Bases de Dados

Ano Letivo 2018/2019

2ª Fase do Projeto

**Cadeia de Hotéis**

**DOCENTE**

Matthias Knorr

**REALIZADO POR**

**Alexandre Correia** (53298)

**Hao Chen** (52636)

**Ricardo Ferreira** (52915)

Turno P5 | Grupo 53

Índice

[**Descrição do Tema**](#_1rxc2u8x8j0g) **3**

[**Modelo ER**](#_vu755k9b98lw) **4**

[**Código SQL**](#_nkzbspp4wbpk) **5**

[**Discussão de limitações/opções tomadas para a implementação da BD**](#_6hqfrb74grd3) **16**

[Espaços](#_cms45g6cydyn) 16

[Reservas](#_ngpjc2i5v1xe) 16

[**Manual do Utilizador**](#_64h0ufc3nt8b) **17**

[Sobre Hotéis](#_berp40qjodf1) 17

[Sobre Funcionários](#_61uafjv65e17) 18

[Sobre Clientes](#_e6wnb7u117j9) 19

[Sobre Espaços](#_4mq9woz51ye6) 19

[Sobre Reservas](#_34cjmhwok2yo) 20

[Sobre Pagamentos](#_2ovk1w5fctbq) 20

# Descrição do Tema

Nesta secção encontra-se uma apresentação e descrição detalhada do nosso tema.

Um grupo de estudantes da FCT pretende fazer um sistema informático para uma cadeia de hotéis. Para tal necessitam de criar uma base de dados com todas as informações dos vários hóteis.

De cada hotel é preciso saber o seu nome, morada, espaços, funcionários e o gestor.

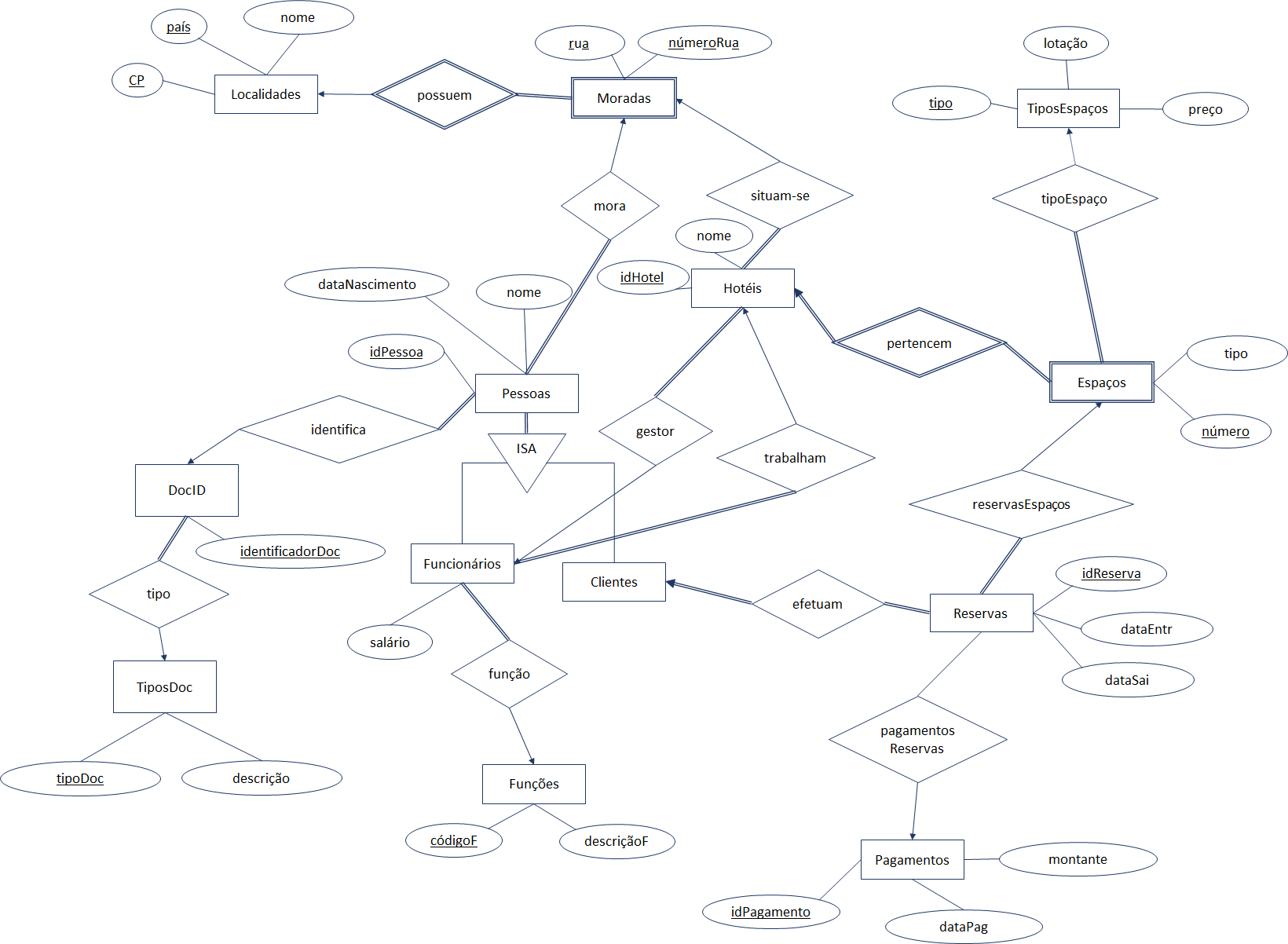
Sabem ainda que cada espaço é identificado por um número e pelo seu hotel onde está registado, tem uma lotação e um preço. Do espaço é preciso saber o tipo.

Acerca de cada funcionário e cliente sabem o seu nome, a sua morada, a sua data de nascimento e um documento de identificação. Contando que os clientes e funcionários podem ser de países diferentes, é necessário saber o tipo do documento de identificação e o identificador do mesmo. E em particular dos funcionários, precisam guardar o seu local de trabalho, ou seja, o hotel em que trabalham e a sua função, de entre gestor, rececionista, limpeza, entre outros, e o salário.

Quanto à reserva, é necessário guardar a data de entrada, o cliente, o espaço e a data de pagamento da mesma**.** Para cada reserva há um único pagamento e só pode ser efetuada por um único cliente.

# Modelo ER

Nesta secção temos o modelo do nosso tema.



**Diferenças:**

* Decidimos que cada reserva apenas deveria aceitar um espaço.
* Passámos a considerar que cada tipo de espaço tinha um preço fixo.

# Código SQL

A aplicação foi criada pela conta Oracle da faculdade.

| drop table localidades cascade constraints;  drop table moradas cascade constraints;  drop table hoteis cascade constraints;  drop table tiposdoc cascade constraints;  drop table docid cascade constraints;  drop table pessoas cascade constraints;  drop table funcoes cascade constraints;  drop table funcionarios cascade constraints;  drop table clientes cascade constraints;  drop table pagamentos cascade constraints;  drop table reservas cascade constraints;  drop table pagamentosreservas cascade constraints;  drop table tiposespacos cascade constraints;  drop table espacos cascade constraints;  drop sequence seq\_id\_reserva;  drop sequence seq\_id\_pessoa;  drop sequence seq\_id\_hotel;  drop sequence seq\_id\_pagamento;  drop sequence seq\_id\_reservas;  drop sequence seq\_id\_espacos;  create table localidades (  CP varchar(20),  pais varchar(100),  nome varchar(100) not null,  primary key (CP, pais)  );  create table moradas (  CP varchar(20),  pais varchar(100),  rua varchar(100),  numeroRua integer,  primary key (CP, pais, rua, numeroRua),  foreign key (CP, pais) references localidades(CP, pais)  );  create table tiposDoc (  tipoDoc char(2),  descricao varchar(25),  primary key (tipoDoc)  );  create table docid (  tipoDoc char(2),  identificadorDoc varchar(15),  primary key (tipoDoc, identificadorDoc),  foreign key (tipoDoc) references tiposDoc(tipoDoc)  );  create table pessoas (  idPessoa integer,  nome varchar(200) not null,  dataNascimento date not null,  CP varchar(20) not null,  pais varchar(100) not null,  rua varchar(100) not null,  numeroRua integer not null,  tipoDoc char(2) not null,  identificadorDoc varchar(15) not null,  primary key (idPessoa),  foreign key (CP, pais, rua, numeroRua) references moradas(CP, pais, rua, numeroRua),  foreign key (tipoDoc, identificadorDoc) references docID(tipoDoc, identificadorDoc)  );  create table funcoes (  codigoF char(2),  descricaoF varchar(20) not null,  primary key (codigoF)  );  create table funcionarios (  idPessoa integer,  salario float not null,  codigoF char(2) not null,  localTrabalho integer not null,  primary key (idPessoa),  foreign key (idPessoa) references pessoas(idPessoa),  foreign key (codigoF) references funcoes(codigoF)  );  create table hoteis (  idHotel integer,  nome varchar(50) not null,  CP varchar(20) not null,  pais varchar(100) not null,  rua varchar(100) not null,  numeroRua integer not null,  gestor integer,  primary key (idHotel),  foreign key (CP, pais, rua, numeroRua) references moradas(CP, pais, rua, numeroRua),  foreign key (gestor) references funcionarios(idPessoa),  constraint morada\_unique unique (CP, pais, rua, numeroRua)  );  alter table funcionarios add constraint fk\_funcionarios foreign key (localTrabalho) references hoteis(idHotel);  create table clientes (  idPessoa integer,  primary key (idPessoa),  foreign key (idPessoa) references pessoas(idPessoa)  );  create table pagamentos (  idPagamento integer,  montante float not null,  dataPag date not null,  primary key (idPagamento)  );  create table tiposespacos (  tipo varchar(20),  lotacao integer not null,  preco float not null,  primary key (tipo)  );  create table espacos (  idHotel integer,  numero varchar(6),  tipo varchar(20) not null,  primary key (idHotel, numero),  foreign key (tipo) references tiposespacos(tipo)  );  create table reservas (  idReserva integer,  dataEntr date not null,  dataSai date not null,  idPessoa integer not null,  idHotel integer not null,  numero varchar(6) not null,  primary key (idReserva),  foreign key (idPessoa) references clientes(idPessoa),  foreign key (idHotel, numero) references espacos(idHotel, numero),  constraint data\_unique unique (idPessoa, dataEntr, dataSai),  constraint check\_date check (dataEntr < dataSai)  );  create table pagamentosreservas (  idReserva integer,  idPagamento integer,  primary key (idReserva, idPagamento),  foreign key (idReserva) references reservas(idReserva),  foreign key (idPagamento) references pagamentos(idPagamento)  );  create sequence seq\_id\_reserva  start with 1  increment by 1;  create sequence seq\_id\_pessoa  start with 1  increment by 1;  create sequence seq\_id\_hotel  start with 1  increment by 1;  create sequence seq\_id\_pagamento  start with 1  increment by 1;  create sequence seq\_id\_reservas  start with 1  increment by 1;  create sequence seq\_id\_espacos  start with 1  increment by 1;    -- TIPOS DOCUMENTO DE IDENTIFICACAO  insert into tiposDoc (tipoDoc, descricao) values ('PS', 'Passaporte');  insert into tiposDoc (tipoDoc, descricao) values ('CC', 'Cartão de Cidadão');  insert into tiposDoc (tipoDoc, descricao) values ('BI', 'Bilhete de Identidade');  -- FUNCOES FUNCIONARIOS  insert into funcoes (codigoF, descricaoF) values ('GT', 'Gestor');  insert into funcoes (codigoF, descricaoF) values ('LP', 'Limpeza');  insert into funcoes (codigoF, descricaoF) values ('SG', 'Segurança');  insert into funcoes (codigoF, descricaoF) values ('CZ', 'Cozinheiro');  insert into funcoes (codigoF, descricaoF) values ('EM', 'Empregado de Mesa');  insert into funcoes (codigoF, descricaoF) values ('RC', 'Receção');  insert into funcoes (codigoF, descricaoF) values ('AM', 'Animador');  -- TIPOS ESPACOS  insert into tiposespacos (tipo, lotacao, preco) values ('Individual', 1, 20);  insert into tiposespacos (tipo, lotacao, preco) values ('Duplo', 2, 40);  insert into tiposespacos (tipo, lotacao, preco) values ('Suíte', 2, 200);  insert into tiposespacos (tipo, lotacao, preco) values ('Salão', 150, 1000);  create or replace view registar\_funcionario as  select idPessoa, dataNascimento, pessoas.nome as nome, rua, numeroRua,  CP, pais, localidades.nome as localidade, tipoDoc, identificadorDoc, salario, codigoF, localTrabalho  from funcionarios inner join pessoas using(idPessoa) inner join moradas using (CP, pais, rua, numeroRua)  inner join localidades using (CP, pais);    create or replace trigger registar\_funcionario\_trigger  instead of insert on registar\_funcionario  declare  c number;  ident number;  begin  select count(\*) into c from hoteis, funcoes where idHotel = :new.localTrabalho and codigoF = :new.codigoF;  if(c > 0) then  select count(\*) into c from localidades where CP = :new.CP and pais = :new.pais;  if(c = 0) then  insert into localidades (CP, pais, nome) values (:new.CP, :new.pais, :new.localidade);  end if;  select count(\*) into c from moradas  where CP = :new.CP and pais = :new.pais and rua = :new.rua and numeroRua = :new.numeroRua;  if(c = 0) then  insert into moradas (CP, pais, rua, numeroRua) values (:new.CP, :new.pais, :new.rua, :new.numeroRua);  end if;  select count(\*) into c from pessoas where tipoDoc = :new.tipoDoc and identificadorDoc = :new.identificadorDoc;  if(c = 0) then  ident := seq\_id\_pessoa.nextval;  insert into docid (tipoDoc, identificadorDoc) values (:new.tipoDoc, :new.identificadorDoc);  insert into pessoas (idPessoa, nome, dataNascimento, CP, pais, rua, numeroRua, tipoDoc, identificadorDoc) values  (ident, :new.nome, :new.dataNascimento, :new.CP, :new.pais, :new.rua, :new.numeroRua,  :new.tipoDoc, :new.identificadorDoc);  end if;  select idPessoa into ident from pessoas where tipoDoc = :new.tipoDoc and identificadorDoc = :new.identificadorDoc;  insert into funcionarios (idPessoa, salario, codigoF, localTrabalho) values  (ident, :new.salario, :new.codigoF, :new.localTrabalho);  select gestor into c from hoteis where idHotel = :new.localTrabalho;  if(:new.codigoF = 'GT' and c is null) then  update hoteis set gestor = ident where idHotel = :new.localTrabalho;  end if;  else  Raise\_Application\_Error (-20100, 'Local de trabalho ou função não existente. Registo não efetuado!');  end if;  end;  /  create or replace trigger editar\_funcionario  instead of update on registar\_funcionario  declare  c number;  begin  update funcionarios set salario = :new.salario, codigoF = :new.codigoF, localTrabalho = :new.localTrabalho  where idPessoa = :old.idPessoa;  update pessoas set nome = :new.nome, dataNascimento = :new.dataNascimento, CP = :new.CP, pais = :new.pais, rua = :new.rua,  numeroRua = :new.numeroRua where idPessoa = :old.idPessoa;  select count(\*) into c from moradas  where CP = :new.CP and pais = :new.pais and rua = :new.rua and numeroRua = :new.numeroRua;  if(c = 0) then  insert into moradas (CP, pais, rua, numeroRua) values (:new.CP, :new.pais, :new.rua, :new.numeroRua);  end if;  select count(\*) into c from localidades where CP = :new.CP and pais = :new.pais;  if(c = 0) then  insert into localidades (CP, pais, nome) values (:new.CP, :new.pais, :new.localidade);  end if;  end;  /  create or replace trigger eliminar\_funcionario  instead of delete on registar\_funcionario  declare  c number;  begin  delete from funcionarios where idPessoa = :old.idPessoa;  delete from docid where tipoDoc = :old.tipoDoc and identificadorDoc = :old.identificadorDoc;  select count(\*) into c from clientes where idPessoa = :old.idPessoa;  if(c = 0) then  delete from pessoas where idPessoa = :old.idPessoa;  end if;  end;  /  create or replace view registar\_cliente as  select idPessoa, dataNascimento, pessoas.nome as nome, rua, numeroRua,  CP, pais, localidades.nome as localidade, tipoDoc, identificadorDoc  from clientes inner join pessoas using (idPessoa) inner join moradas using (CP, pais, rua, numeroRua) inner join localidades using (CP, pais);    create or replace trigger registar\_cliente\_trigger  instead of insert on registar\_cliente  declare  c number;  ident number;  begin  select count(\*) into c from localidades where CP = :new.CP and pais = :new.pais;  if(c = 0) then  insert into localidades (CP, pais, nome) values (:new.CP, :new.pais, :new.localidade);  end if;  select count(\*) into c from moradas where CP = :new.CP and pais = :new.pais and rua = :new.rua and numeroRua = :new.numeroRua;  if(c = 0) then  insert into moradas (CP, pais, rua, numeroRua) values (:new.CP, :new.pais, :new.rua, :new.numeroRua);  end if;  select count(\*) into c from pessoas where tipoDoc = :new.tipoDoc and identificadorDoc = :new.identificadorDoc;  if( c = 0) then  ident := seq\_id\_pessoa.nextval;  insert into docid (tipoDoc, identificadorDoc) values (:new.tipoDoc, :new.identificadorDoc);  insert into pessoas (idPessoa, nome, dataNascimento, CP, pais, rua, numeroRua, tipoDoc, identificadorDoc) values  (ident, :new.nome, :new.dataNascimento, :new.CP, :new.pais, :new.rua, :new.numeroRua,  :new.tipoDoc, :new.identificadorDoc);  end if;  select idPessoa into ident from pessoas where tipoDoc = :new.tipoDoc and identificadorDoc = :new.identificadorDoc;  insert into clientes (idPessoa) values (ident);  end;  /  create or replace trigger editar\_cliente  instead of update on registar\_cliente  declare  c number;  begin  update pessoas set nome = :new.nome, dataNascimento = :new.dataNascimento, CP = :new.CP, pais = :new.pais, rua = :new.rua,  numeroRua = :new.numeroRua where idPessoa = :old.idPessoa;  select count(\*) into c from moradas  where CP = :new.CP and pais = :new.pais and rua = :new.rua and numeroRua = :new.numeroRua;  if(c = 0) then  insert into moradas (CP, pais, rua, numeroRua) values (:new.CP, :new.pais, :new.rua, :new.numeroRua);  end if;  select count(\*) into c from localidades where CP = :new.CP and pais = :new.pais;  if(c = 0) then  insert into localidades (CP, pais, nome) values (:new.CP, :new.pais, :new.localidade);  end if;  end;  /  create or replace trigger eliminar\_cliente  instead of delete on registar\_cliente  declare  c number;  begin  delete from clientes where idPessoa = :old.idPessoa;  delete from docid where tipoDoc = :old.tipoDoc and identificadorDoc = :old.identificadorDoc;  select count(\*) into c from funcionarios where idPessoa = :old.idPessoa;  if(c = 0) then  delete from pessoas where idPessoa = :old.idPessoa;  end if;  end;  /  create or replace view registar\_hotel as  select hoteis.nome as nome, rua, numeroRua, CP, pais, localidades.nome as localidade  from hoteis inner join moradas using (rua, numeroRua, CP, pais) inner join localidades using (CP, pais);  create or replace trigger registar\_hotel\_trigger  instead of insert on registar\_hotel  declare  c number;  ident number := seq\_id\_hotel.nextval;  begin  select count(\*) into c from localidades where CP = :new.CP and pais = :new.pais;  if (c = 0) then  insert into localidades (CP, pais, nome) values (:new.CP, :new.pais, :new.localidade);  end if;  select count (\*) into c from moradas where CP = :new.CP and pais = :new.pais and rua = :new.rua and  numeroRua = :new.numeroRua;  if (c = 0) then  insert into moradas (CP, pais, rua, numeroRua) values (:new.CP, :new.pais, :new.rua, :new.numeroRua);  end if;  insert into hoteis (idHotel, nome, CP, pais, rua, numeroRua, gestor) values (ident, :new.nome, :new.CP, :new.pais,  :new.rua, :new.numeroRua, null);  end;  /  create or replace view introduzir\_reserva as  select idReserva, dataEntr, dataSai, tipoDoc, identificadorDoc, idHotel, numero  from espacos inner join reservas using (idHotel, numero) inner join clientes using (idPessoa) inner join pessoas using (idPessoa);  create or replace trigger introduzir\_reserva\_trigger  instead of insert on introduzir\_reserva  declare  ident number;  idR number := seq\_id\_reserva.nextval;  begin  select idPessoa into ident from pessoas where tipoDoc = :new.tipoDoc and identificadorDoc = :new.identificadorDoc;  insert into reservas (idReserva, dataEntr, dataSai, idPessoa, idHotel, numero)  values(idR, :new.dataEntr, :new.dataSai, ident, :new.idHotel, :new.numero);  end;  /  create or replace trigger apagar\_reserva\_trigger  instead of delete on introduzir\_reserva  declare  c number;  begin  select count(\*) into c from pagamentosreservas where idReserva = :old.idReserva;  if (c > 0)  then Raise\_Application\_Error (-20100, 'Reserva com pagamento já efetuado. Reserva não eliminada!');  end if;  delete from reservas where idReserva = :old.idReserva;  end;  /  create or replace view registar\_pagamento as  select idPagamento, idReserva, montante, dataEntr, dataSai, tipoDoc, identificadorDoc  from pagamentos inner join pagamentosreservas using (idPagamento) inner join reservas using (idReserva)  inner join pessoas using (idPessoa);  create or replace trigger registar\_pagamento\_trigger  instead of insert on registar\_pagamento  declare  c number;  p number;  idR number;  ident number;  pag number := seq\_id\_pagamento.nextval;  begin  select idPessoa into ident from pessoas where tipoDoc = :new.tipoDoc and identificadorDoc = :new.identificadorDoc;  select idReserva into idR from reservas where idPessoa = ident and dataEntr = :new.dataEntr and dataSai = :new.dataSai;  insert into pagamentos (idPagamento, montante, dataPag) values (pag, :new.montante, :new.dataSai);  /\*select count(\*) into c from pagamentosreservas where idReserva = idR;  select preco \* (dataSai - dataEntr) into p from pessoas inner join reservas using (idPessoa) inner join espacos using (idHotel, numero) inner join tipoespacos using (tipo)  where dataEntr = :new.dataEntr and dataSai = :new.dataSai and tipoDoc = :new.tipoDoc and identificadorDoc = :new.identificadorDoc;  if(c = 0 and p = :new.montante) then  insert into pagamentosreservas (idReserva, idPagamento) values (idR, pag);  end if;  if (c = 0 and p != :new.montante) then  Raise\_Application\_Error (-20100, 'O valor do montante está incorreto. Pagamento não efetuado!');  else  Raise\_Application\_Error (-20100, 'Reserva com pagamento já efetuado. Pagamento não aceite!');  end if;\*/  end;  /  -- ESPACOS  insert into registar\_espaco (idHotel, numero, tipo) values (1, '1.01', 'Individual');  insert into registar\_espaco (idHotel, numero, tipo) values (1, '2.01', 'Duplo');  insert into registar\_espaco (idHotel, numero, tipo) values (1, '3.01', 'Suíte');  insert into registar\_espaco (idHotel, numero, tipo) values (1, '100', 'Salão');  -- TESTES  -- hoteis  insert into registar\_hotel (nome, rua, numeroRua, CP, pais, localidade) values  ('Alexandre', 'Rua 1 de Maio', 10, '2222-333', 'Portugal', 'Almada');  -- funcionarios  insert into registar\_funcionario (idPessoa, dataNascimento, nome, rua, numeroRua,  CP, pais, localidade, tipoDoc, identificadorDoc, salario, codigoF, localTrabalho) values  (1, '04-Dec-98', 'Hao', 'Rua de Teste', 1234, '1234-567', 'Portugal', 'Almada', 'CC', '123456789', 10000, 'GT', 1);  insert into registar\_funcionario (idPessoa, dataNascimento, nome, rua, numeroRua,  CP, pais, localidade, tipoDoc, identificadorDoc, salario, codigoF, localTrabalho) values  (2, '04-Dec-98', 'Hao2', 'Rua de Teste', 1234, '1234-567', 'Portugal', 'Almada', 'CC', '123456788', 10000, 'RC', 1);  -- clientes  insert into registar\_cliente (idPessoa, dataNascimento, nome, rua, numeroRua, CP, pais, localidade, tipoDoc, identificadorDoc) values  (3, '04-Dec-98', 'Alexandre', 'Rua de Teste', 4, '2222-333', 'Portugal', 'Almada', 'CC', '123456782');  insert into registar\_cliente (idPessoa, dataNascimento, nome, rua, numeroRua, CP, pais, localidade, tipoDoc, identificadorDoc) values  (4, '04-Dec-98', 'Alexandre2', 'Rua de Teste', 4, '2222-333', 'Portugal', 'Almada', 'CC', '234567891');  -- reservas  insert into introduzir\_reserva (idReserva, dataEntr, dataSai, tipoDoc, identificadorDoc, idHotel, numero)  values (1, '01-May-19', '03-May-19', 'CC', '123456782', 1, '1.01');  insert into introduzir\_reserva (idReserva, dataEntr, dataSai, tipoDoc, identificadorDoc, idHotel, numero)  values (2, '01-May-19', '03-May-19', 'CC', '234567891', 1, '2.01');  -- reservas espacos  /\*insert into reservasespacos (idHotel, numero, idReserva) values (1, '1.01', 1);  insert into reservasespacos (idHotel, numero, idReserva) values (1, '2.01', 1);\*/  commit; |
| --- |

Os triggers de registar e introduzir servem para colocar as informações passadas nas vistas nas tabelas da base de dados referentes às informações.

Os triggers de editar e apagar servem para editar ou apagar, respetivamente, as informações nas tabelas respetivas dos elementos inseridos e processados pelos triggers referidos em cima.

# Discussão de limitações/opções tomadas para a implementação da BD

Nesta secção abordamos todas as limitações e consequentemente as opções tomadas para contornar essas limitações.

### Espaços

Ao tentarmos mostrar o nome do hotel em vez do seu id (número interno), aparece bem na página que lista todos os espaços, mas quando tentamos ver as informações do espaço, o Apex dá o erro.

Para contornar esta situação decidimos na página que mostra todos os espaços existentes mostrar o id do hotel em vez do seu nome e assim é possível criar e ver as informações dos espaços, aparecendo o nome do hotel via uma LoV (List of Values).

### Reservas

Ao tentarmos fazer a página das Reservas usando um Master Detail, na implementação do Detail tivemos problemas quanto à partilha de informações entre o Master e o Detail.

Para contornar esta situação decidimos que cada reserva só pode ter um espaço associado.

# Manual do Utilizador

Nesta secção damos uma explicação de como usar cada funcionalidade implementada.

Para aceder às várias funcionalidades clicamos, no menu lateral esquerdo, na opção pretendida.

### Sobre Hotéis

* Listagem

Ao aceder a esta funcionalidade, aparece uma lista de todos os hotéis registados.

* Inserção

Na página onde aparece a lista de todos os hotéis, clicamos em “Create” (canto superior direito) e aparece uma outra página com título “Registar Hotel”. Nessa página, depois de preencher todos os campos necessários, clicamos no botão “Create” (canto inferior direito), criando assim um novo hotel.

.

* Eliminação

Na página onde temos a lista de todos os hotéis, clicamos no lápis que aparece na primeira coluna da linha do hotel que pretendemos eliminar e aparece uma outra página com título “Registar Hotel**”.** Nessa página, ao clicarmos em “Delete” (canto inferior direito), eliminamos o hotel.

* Alteração

Na página onde podemos ver a lista de todos os hotéis, clicamos no lápis que aparece na primeira coluna da linha do hotel que pretendemos editar e assim aparece uma outra página com título “Registar Hotel”**.** Nessa página alteramos os dados que pretendemos alterar e clicamos em “Apply Changes” (canto inferior direito), alterando assim as informações do hotel.

### Sobre Funcionários

* Listagem

Acedendo a esta funcionalidade, podemos ver as várias informações de cada funcionário de cada hotel.

* Inserção

Na página onde aparece a lista de todos os funcionários, clicamos em “Create” (canto superior direito) e aparece uma outra página com título “Registar Funcionário”.

Depois de preencher todos os campos, clicamos no botão “Create” (canto inferior direito), registando assim um novo funcionário.

* Eliminação

Na página onde temos a lista de todos os funcionários, clicamos no lápis que aparece na primeira coluna da linha do funcionário que pretendemos eliminar. Abrimos assim uma outra página com título “Registar Funcionário**”.**

Ao clicarmos em “Delete” (canto inferior direito), o funcionário é eliminado.

* Alteração

Para editar as informações de um funcionário clicamos no lápis à esquerda das informações do funcionário, temos de estar na página onde podemos ver a lista de todos os funcionários, clicamos no lápis que aparece na primeira coluna da linha do funcionário que pretendemos editar e assim aparece uma outra página com título “Registar Funcionário”**.** Nessa página, alteramos os dados que pretendemos alterar e clicamos em “Apply Changes” (canto inferior direito), alterando assim as informações do funcionário.

### Sobre Clientes

* Listagem

A lista de todos os clientes registados no sistema aparece diretamente quando acedemos a esta funcionalidade.

* Inserção

Na página onde aparece a lista de todos os clientes, clicamos em “Create” (canto superior direito) e aparece uma outra página com título “Registar Cliente”. Nessa página, depois de preencher todos os campos necessários, clicamos em “Create” (canto inferior direito), inserindo assim um novo cliente.

* Eliminação

Um cliente não pode ser apagado.

* Alteração

Para editar as informações de um cliente temos de estar na página onde podemos ver a lista de todos os clientes, clicamos no lápis que aparece na primeira coluna da linha do cliente que pretendemos editar e assim aparece uma outra página com título “Registar Cliente”**.**

Alteramos os dados que pretendemos alterar e clicamos em “Apply Changes” (canto inferior direito), alterando assim as informações do cliente.

### Sobre Espaços

* Listagem

Acedendo a esta funcionalidade, vemos uma lista de todos os espaços registados de cada hotel.

* Inserção

Na página onde aparece a lista de todos os espaços, clicamos em “Create” (canto superior direito) e aparece uma outra página com título “Inserir Novo Espaço”. Nessa página, depois de preencher todos os campos necessários, clicamos em “Create” (canto inferior direito), criando assim um novo espaço.

* Eliminação

Na página onde temos a lista de todos os espaços, clicamos no lápis que aparece na primeira coluna da linha do espaço que pretendemos eliminar e aparece uma outra página com título “Inserir Novo Espaço”. Nessa página, ao clicarmos em “Delete” (canto inferior direito), o espaço é eliminado.

* Alteração

Não é possível efetuar alterações aos espaços.

### Sobre Reservas

* Listagem

A lista de todas as reservas registadas aparece quando acedemos a esta funcionalidade.

* Inserção

Na página principal das reservas, clicamos em “Create” (canto superior direito) e aparece uma outra página com título “Check-In”.

Depois de preencher todos os campos, clicamos no botão “Create” (canto inferior direito), registando assim uma nova reserva.

* Eliminação

Só é possível eliminar uma reserva se ainda não tiverem sido feitos pagamentos da mesma.

* Alteração

Não é possível alterar uma reserva.

### Sobre Pagamentos

* Listagem

A lista de todos os pagamentos efetuados aparece quando entramos nesta funcionalidade.

* Inserção

Na página principal dos pagamentos, clicamos em “Create” (canto superior direito) e aparece uma outra página com título “Check-Out”.

Depois de preencher todos os campos, clicamos no botão “Create” (canto inferior direito), registrando assim um novo pagamento.

* Eliminação

Não se pode eliminar um pagamento.

* Alteração

Não é possível alterar um pagamento.