Desenvolvimento de uma Base de Dados para uma Companhia Aérea

Bases de Dados - 2015/2016

João Ferreira up201405163@fe.up.pt
Nuno Silva up201404380@fe.up.pt
Pedro Amaro up201405163@fe.up.pt

Dirigido por:

Carla Alexandra Teixeira Lopes ctl@fe.up.pt

Índice

Introdução

Este relatório foi feito no âmbito da unidade curricular "Base de Dados" e centra-se no estudo da melhor maneira de criar uma base de dados, para uma companhia aérea, na perspetiva do utilizador. Com esta base de dados, o utilizador poderá consultar informações sobre os voos disponíveis, ver o preço de uma reserva e ajustar várias opções em relação aos voos de cada passageiro.

Descrição do contexto

Esta base de dados foca-se no armazenamento de informação relevante para um utilizador. Será possível gerir a oferta de voos, tal como todos os detalhes dos voos em questão (origem, destino e lugares disponíveis). Um utilizador, ao fazer uma reserva, que pode ser constituída por vários voos, necessita de indicar os passageiros pertencentes à mesma. Para além disso, o utilizador terá oportunidade de fazer escolhas sobre o voo de cada passageiro (refeição predileta, refeições extra, malas e animais de estimação).

Surgiu um impasse com o cálculo do preço, sendo que foi recomendado o desenvolvimento e a explicitação de um algoritmo exemplificativo que calculasse o preço da reserva.

Definições dos principais conceitos

Uma reserva é constituída por passageiros e vários voos, numa certa ordem, tendo um preço associado, que varia com vários fatores, nomeadamente se a reserva for de ida e de volta (se a origem do primeiro voo e o destino do ultimo coincidirem).

Cada voo, associado a um trajeto, tem uma lotação limitada de passageiros, assim como datas e horas de partida/chegada referentes ao mesmo. Um trajeto tem um nome identificativo, um aeroporto de partida e outro de chegada (por exemplo, “TP0087” seria entre Lisboa e São Paulo).

Um passageiro, em cada voo, pode escolher as refeições que irá consumir, assim como a sua quantidade, a sua classe, e pode, inclusive, levar no porão do avião malas *standard* (até 23kg, com dimensões menores que 20cm x 30cm x 50cm) ou malas não *standard* (até 32kg, com dimensões menores que 50cm x 75cm x 150cm), assim como até 2 animais de estimação.

Finalmente, cada aeroporto serve uma cidade, num dado país e possui um código identificativo de 3 caracteres (código IATA).

Cálculo do preço da reserva

Como foi dito anteriormente, não é possível conhecer a equação exata usada pelas companhias aéreas para calcular o preço das suas reservas, tornando-se necessário assumir uma expressão intuitivamente correta para o cálculo do preço de cada reserva. Devido à dificuldade deste problema, foi decidido dedicar uma secção à descrição do algoritmo desenvolvido.

Sendo $mRetorno$ o modificador do preço de retorno (será 1 se a reserva for apenas de ida, ou 0,7 se for de ida e de volta), $nVoos$ o número de voos, $nPax$ o número de passageiros e $Pvoo(i, j)$ o preço do voo i para o passageiro j, o preço de uma reserva pode ser dado por:

$$Preserva = mRetorno \sum_{i=0}^{nVoos} \sum_{j=0}^{nPax} Pvoo(i, j)$$

O preço de um voo i, para um passageiro j, é calculado tendo em conta um modificador de classe $mClasse(i, j)$ (1 para a classe turística e 3 para a primeira classe), um modificador baseado no número de passageiros na reserva $mColectivo(j)$, sendo que cada passageiro adicional na reserva significa uma diminuição no preço do voo, segundo a fórmula:

$$mColectivo(j) = 5/16 * (4/5)^{j} + 3/4$$

Tendo ainda em conta o preço base do voo $Pbase(i)$ e o preço de qualquer extra que o passageiro tenha comprado para este voo $Pextra(i, j)$, o preço do voo é calculado da seguinte forma:

$$Pvoo(i, j) = mClasse(i, j)\, mColectivo(j)\, Pbase(i, d, p) + Pextra(i, j)$$

O preço base de um voo i é baseado na distância $dist(i)$ deste e em 3 modificadores: $mDias(d)$, $mCheio(p)$ e $mDiaEspecial$. Os primeiros dois podem ser calculados da seguinte forma: sendo d o número de dias restantes até à data de partida do voo e p a percentagem de ocupação da lotação atual do avião:

$$mDias(d) = 1 + 6,75 * d^{-2}$$

$$mCheio(p) = \frac{2}{5}(4 - 3(1 - p))$$

O modificador $mDiaEspecial$ é 1,5 se a data de partida do voo coincidir com um feriado nacional e 1 nos outros dias. Assim pode-se calcular o preço base:

$$Pbase(i, d, p) = (35 + 0,08\ dist(i))\ mDiaEspecial\ mCheio(p)\ mDias(d)$$

Finalmente, o preço extra de cada passageiro j num voo i são os gastos extras nos seguintes campos: malas, animais de estimação e refeições, e pode ser calculado da seguinte forma:

$$Pextra(i, j) = Pmalas(i, j) + Panimais(i, j) + Prefeicao(i, j)$$

Os campos à direita desta equação podem ser determinados segundo os seguintes critérios:

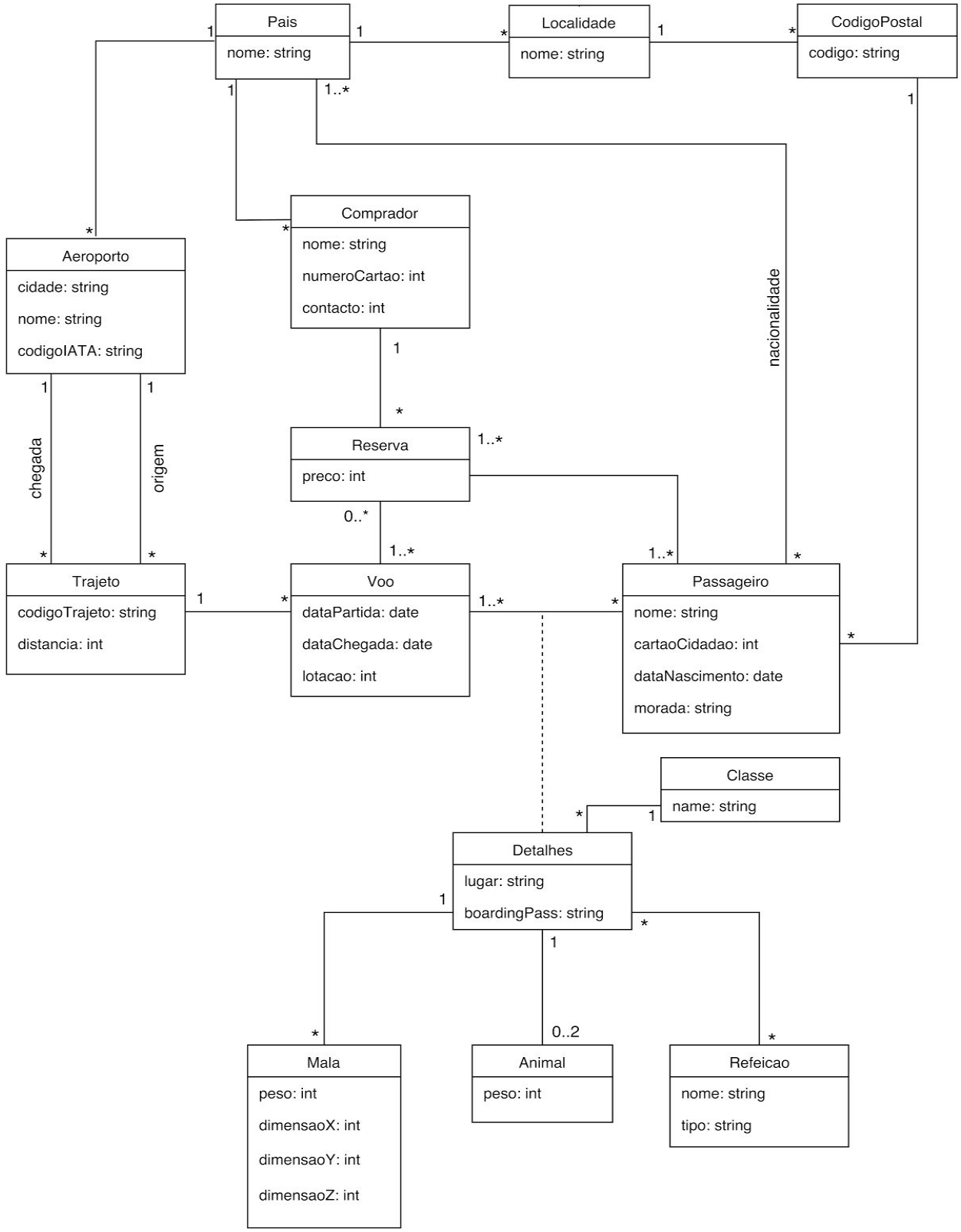
Preço de malas *standard*					
N	1	2	3	4	n
Preço / voo	0	20	50	100	+100 / mala

Os preços de malas não *standard* são determinados a partir da soma do preço por volume (1500/m^3) e do preço por peso (2/kg).

Preço para animais		
N	1	2
Preço / voo	50	150

Preço das refeiçoes			
N	1	2	n
Preço / voo	0	5	+10 / refeição

Diagrama UML



Esquema Relacional

Pais(idPais, nome)

Localidade(idLocalidade, idPais→Pais, nome)

CodigoPostal(idCodigoPostal, idLocalidade→Localidade, codigo)

Comprador(idComprador, nome, numeroCartao, contato, idPais→Pais)

Reserva(idReserva, preco, idComprador→Comprador)

Aeroporto(idAeroporto, cidade, nome, codigoIATA, idPais→Pais)

Trajeto(idTrajeto, idAeroportoOrigem→Aeroporto, idAeroportoChegada→Aeroporto, codigoTrajeto, distancia)

Voo(idVoo, idTrajeto→Trajeto, dataPartida, dataChegada, lotacao)

Passageiro(idPassageiro, nome, cartaoCidadao, dataNascimento, morada, idCodigoPostal→CodigoPostal)

PassageiroNacionalidade(idPassageiro→Passageiro, idPais→Pais)

Classe(idClasse, nome)

Mala(idMala, idVoo→Voo, idPassageiro→Passageiro, peso, dimensaoX, dimensaoY, dimensaoZ)

Animal(idAnimal, idVoo→Voo, idPassageiro→Passageiro, peso)

Refeicao(idRefeicao, nome, tipo)

Detalhes(idVoo→Voo, idPassageiro→Passageiro, lugar, boardingPass, idClasse→Classe)

ReservaPassageiro(idPassageiro→Passageiro, idReserva→Reserva)

ReservaVoo(idVoo→Voo, idReserva→Reserva)

DetalhesRefeicao(idVoo→Voo, idPassageiro→Passageiro, idRefeicao→Refeicao)

Instruções LDD-SQL para a criação da Base de Dados

```
PRAGMA FOREIGN_KEYS=ON;

.mode columns
.headers on
.nullvalue NULL

DROP TABLE IF EXISTS Pais;
CREATE TABLE Pais (
       idPais NUMBER PRIMARY KEY,
       nome TEXT
       );

DROP TABLE IF EXISTS Localidade;
CREATE TABLE Localidade (
       idLocalidade NUMBER PRIMARY KEY,
       idPais NUMBER,
       nome TEXT,
       FOREIGN KEY(idPais) REFERENCES Pais(idPais)
       );

DROP TABLE IF EXISTS CodigoPostal;
CREATE TABLE CodigoPostal (
       idCodigoPostal NUMBER PRIMARY KEY,
       idLocalidade NUMBER,
       codigo TEXT,
       FOREIGN KEY(idLocalidade) REFERENCES Localidade(idLocalidade)
       );

DROP TABLE IF EXISTS Comprador;
CREATE TABLE Comprador (
       idComprador NUMBER PRIMARY KEY,
       nome TEXT,
       numeroCartao NUMBER,
       contato NUMBER,
       idPais NUMBER,
       FOREIGN KEY(idPais) REFERENCES Pais(idPais)
       );

DROP TABLE IF EXISTS Reserva;
CREATE TABLE Reserva (
       idReserva NUMBER PRIMARY KEY,
       preco NUMBER,
       idComprador NUMBER,
       FOREIGN KEY(idComprador) REFERENCES Comprador(idComprador)
       );

DROP TABLE IF EXISTS Aeroporto;
CREATE TABLE Aeroporto (
       idAeroporto NUMBER PRIMARY KEY,
       cidade TEXT,
       nome TEXT,
       codigoIATA TEXT,
       idPais NUMBER,
       FOREIGN KEY(idPais) REFERENCES Pais(idPais)
       );
```

```
DROP TABLE IF EXISTS Trajeto;
CREATE TABLE Trajeto (
       idTrajeto NUMBER PRIMARY KEY,
       idAeroportoOrigem NUMBER,
       idAeroportoChegada NUMBER,
       codigoTrajeto TEXT,
       distancia NUMBER,
       FOREIGN KEY(idAeroportoOrigem) REFERENCES Aeroporto(idAeroporto),
       FOREIGN KEY(idAeroportoChegada) REFERENCES Aeroporto(idAeroporto)
       );

DROP TABLE IF EXISTS Voo;
CREATE TABLE Voo (
       idVoo NUMBER PRIMARY KEY,
       idTrajeto NUMBER,
       dataPartida DATE,
       dataChegada DATE,
       lotacao NUMBER,
       FOREIGN KEY(idTrajeto) REFERENCES Trajeto(idTrajeto)
       );

DROP TABLE IF EXISTS Passageiro;
CREATE TABLE Passageiro (
       idPassageiro NUMBER PRIMARY KEY,
       nome TEXT,
       cartaoCidadao NUMBER,
       dataNascimento DATE,
       morada TEXT,
       idCodigoPostal NUMBER,
       FOREIGN KEY(idCodigoPostal) REFERENCES CodigoPostal(idCodigoPostal)
       );

DROP TABLE IF EXISTS PassageiroNacionalidade;
CREATE TABLE PassageiroNacionalidade (
       idPassageiro NUMBER NOT NULL,
       idPais NUMBER NOT NULL,
       FOREIGN KEY(idPassageiro) REFERENCES Passageiro(idPassageiro),
       FOREIGN KEY(idPais) REFERENCES Pais(idPais),
       CONSTRAINT idPassageiroNacionalidade PRIMARY KEY (idPassageiro, idPais)
       );

DROP TABLE IF EXISTS Classe;
CREATE TABLE Classe (
       idClasse NUMBER PRIMARY KEY,
       nome TEXT
       );

DROP TABLE IF EXISTS Mala;
CREATE TABLE Mala (
       idMala NUMBER NOT NULL,
       idVoo NUMBER NOT NULL,
       idPassageiro NUMBER NOT NULL,
       peso NUMBER,
       dimensaoX NUMBER,
       dimensaoY NUMBER,
       dimensaoZ NUMBER,
       FOREIGN KEY(idVoo) REFERENCES Voo(idVoo),
       FOREIGN KEY(idPassageiro) REFERENCES Passageiro(idPassageiro),
       CONSTRAINT idMalaVooPassageiro PRIMARY KEY (idMala, idVoo, idPassageiro)
       );
```

```
DROP TABLE IF EXISTS Animal;
CREATE TABLE Animal (
       idAnimal NUMBER NOT NULL,
       idVoo NUMBER NOT NULL,
       idPassageiro NUMBER NOT NULL,
       peso NUMBER,
       FOREIGN KEY(idVoo) REFERENCES Voo(idVoo),
       FOREIGN KEY(idPassageiro) REFERENCES Passageiro(idPassageiro),
       CONSTRAINT idAnimalVooPassageiro PRIMARY KEY (idAnimal, idVoo, idPassageiro)
       );

DROP TABLE IF EXISTS Refeicao;
CREATE TABLE Refeicao (
       idRefeicao NUMBER PRIMARY KEY,
       nome TEXT,
       tipo TEXT
       );

DROP TABLE IF EXISTS Detalhes;
CREATE TABLE Detalhes (
       idVoo NUMBER NOT NULL,
       idPassageiro NUMBER NOT NULL,
       lugar TEXT,
       boardingPass TEXT,
       idClasse NUMBER,
       FOREIGN KEY(idVoo) REFERENCES Voo(idVoo),
       FOREIGN KEY(idPassageiro) REFERENCES Passageiro(idPassageiro),
       FOREIGN KEY(idClasse) REFERENCES Classe(idClasse),
       CONSTRAINT idDetalhes PRIMARY KEY (idVoo, idPassageiro)
       );

DROP TABLE IF EXISTS ReservaPassageiro;
CREATE TABLE ReservaPassageiro (
       idPassageiro NUMBER NOT NULL,
       idReserva NUMBER NOT NULL,
       FOREIGN KEY(idPassageiro) REFERENCES Passageiro(idPassageiro),
       FOREIGN KEY(idReserva) REFERENCES Reserva(idReserva),
       CONSTRAINT idReservaPassageiro PRIMARY KEY (idPassageiro, idReserva)
       );

DROP TABLE IF EXISTS ReservaVoo;
CREATE TABLE ReservaVoo (
       idVoo NUMBER NOT NULL,
       idReserva NUMBER NOT NULL,
       FOREIGN KEY(idVoo) REFERENCES Voo(idVoo),
       FOREIGN KEY(idReserva) REFERENCES Reserva(idReserva),
       CONSTRAINT idReservaVoo PRIMARY KEY (idVoo, idReserva)
       );

DROP TABLE IF EXISTS DetalhesRefeicao;
CREATE TABLE DetalhesRefeicao (
       idVoo NUMBER NOT NULL,
       idPassageiro NUMBER NOT NULL,
       idRefeicao NUMBER NOT NULL,
       FOREIGN KEY(idVoo) REFERENCES Voo(idVoo),
       FOREIGN KEY(idPassageiro) REFERENCES Passageiro(idPassageiro),
       FOREIGN KEY(idRefeicao) REFERENCES Refeicao(idRefeicao)
       CONSTRAINT idVooPassageiroRefeicao PRIMARY KEY (idVoo, idPassageiro,
idRefeicao)
       );
```

Instruções LMD-SQL para o preenchimento da Base de Dados

```
PRAGMA FOREIGN_KEYS=ON;

.mode columns
.headers on
.nullvalue NULL

INSERT INTO Classe VALUES (0, "Turistica");
INSERT INTO Classe VALUES (1, "Primeira Classe");

INSERT INTO Refeicao VALUES (0, "Galinha", "Carne");
INSERT INTO Refeicao VALUES (1, "Bife", "Carne");
INSERT INTO Refeicao VALUES (2, "Bacalhau", "Peixe");
INSERT INTO Refeicao VALUES (3, "Salada", "Vegetariana");

INSERT INTO Pais VALUES ( 0, "Portugal");
INSERT INTO Pais VALUES ( 1, "Espanha");
INSERT INTO Pais VALUES ( 2, "Franca");
INSERT INTO Pais VALUES ( 3, "Belgica");
INSERT INTO Pais VALUES ( 4, "Reino Unido");
INSERT INTO Pais VALUES ( 5, "Brazil");
INSERT INTO Pais VALUES ( 6, "Russia");
INSERT INTO Pais VALUES ( 7, "Polonia");
INSERT INTO Pais VALUES ( 8, "Republica Checa");
INSERT INTO Pais VALUES ( 9, "Gana");
INSERT INTO Pais VALUES (10, "Estados Unidos");
INSERT INTO Pais VALUES (11, "Angola");

INSERT INTO Aeroporto VALUES ( 0, "Madrid", "Madrid-Barajas", "MAD", 1);
INSERT INTO Aeroporto VALUES ( 1, "Lisboa", "Portela", "LIS", 0);
INSERT INTO Aeroporto VALUES ( 2, "Barcelona", "Barcelona", "BCN", 1);
INSERT INTO Aeroporto VALUES ( 3, "Recife", "Int'l do Recife", "REC", 5);
INSERT INTO Aeroporto VALUES ( 4, "Moscovo", "Int'l Domodedovo", "DME", 6);
INSERT INTO Aeroporto VALUES ( 5, "Warsaw", "Frederic Chopin de Varsovia", "WAW",
7);
INSERT INTO Aeroporto VALUES ( 6, "Praga", "Int'l Ruzyne", "PRG", 8);
INSERT INTO Aeroporto VALUES ( 7, "Acra", "Int'l Kotoka", "ACC", 9);
INSERT INTO Aeroporto VALUES ( 8, "Santa Cruz", "Madeira", "FNC", 0);
INSERT INTO Aeroporto VALUES ( 9, "Lajes", "Base Aerea das Lajes", "TER", 0);
INSERT INTO Aeroporto VALUES (10, "Newark", "Int'l de Newark", "KEWR", 10);
INSERT INTO Aeroporto VALUES (11, "Miami", "Int'l de Miami", "KMIA", 10);
INSERT INTO Aeroporto VALUES (12, "Luanda", "Quatro de Fevereiro", "LAD", 11);
INSERT INTO Aeroporto VALUES (13, "Fortaleza", "Int'l de Fortaleza", "FOR", 5);
INSERT INTO Aeroporto VALUES (14, "Manchester", "Manchester", "MAN", 4);
INSERT INTO Aeroporto VALUES (15, "Oostende", "Ostend-Bruges Int'l", "OST", 3);
INSERT INTO Aeroporto VALUES (16, "Londres", "Londres Heathrow", "LHR", 4);
INSERT INTO Aeroporto VALUES (17, "Paris", "Paris-Orly", "ORY", 2);
INSERT INTO Aeroporto VALUES (18, "Porto", "Francisco Sa Carneiro", "OPO", 0);

INSERT INTO Trajeto VALUES ( 0,  0,  1, "TAP1017", 514);
INSERT INTO Trajeto VALUES ( 1,  1,  0, "TAP1018", 512);
INSERT INTO Trajeto VALUES ( 2,  2,  1, "TAP1045", 995);
INSERT INTO Trajeto VALUES ( 3,  1,  3, "TAP11"  , 5864);
INSERT INTO Trajeto VALUES ( 4,  1,  4, "TAP1232", 3918);
INSERT INTO Trajeto VALUES ( 5,  5,  1, "TAP1261", 2752);
INSERT INTO Trajeto VALUES ( 6,  6,  1, "TAP1305", 2231);
INSERT INTO Trajeto VALUES ( 7,  1,  7, "TAP1511", 3802);
```

```
INSERT INTO Trajeto VALUES ( 8,  8, 18, "TAP1714", 1192);
INSERT INTO Trajeto VALUES ( 9,  1,  9, "TAP1823", 1556);
INSERT INTO Trajeto VALUES (10,  1, 10, "TAP203" , 5439);
INSERT INTO Trajeto VALUES (11,  1, 12, "TAP287" , 5787);
INSERT INTO Trajeto VALUES (12,  1, 13, "TAP31"  , 5618);
INSERT INTO Trajeto VALUES (13, 15,  1, "TAP3383", 1670);
INSERT INTO Trajeto VALUES (14, 16, 16, "TAP367" , 1566);
INSERT INTO Trajeto VALUES (15,  1, 17, "TAP448" , 1439);
INSERT INTO Trajeto VALUES (16, 17,  1, "TAP449" , 1439);
INSERT INTO Trajeto VALUES (17, 17, 18, "TAP457" , 1201);
INSERT INTO Trajeto VALUES (18, 18, 17, "TAP458" , 1201);

INSERT INTO Voo VALUES (0,  12, "2016-04-25 14:34", "2016-04-25 17:38", 200);
INSERT INTO Voo VALUES (1,  4,  "2016-04-25 19:28", "2016-04-26 02:08", 200);
INSERT INTO Voo VALUES (2,  0,  "2016-04-25 19:20", "2016-04-25 19:15", 150);
INSERT INTO Voo VALUES (3,  3,  "2016-04-25 13:19", "2016-04-25 16:49", 320);
INSERT INTO Voo VALUES (4,  10, "2016-04-25 13:03", "2016-04-25 15:13", 250);
INSERT INTO Voo VALUES (5,  11, "2016-04-26 13:13", "2016-04-26 20:54", 140);
INSERT INTO Voo VALUES (6,  13, "2016-04-25 18:40", "2016-04-25 19:45", 200);
INSERT INTO Voo VALUES (7,  18, "2016-04-23 19:45", "2016-04-23 22:35", 220);
INSERT INTO Voo VALUES (8,  16, "2016-04-25 20:44", "2016-04-25 22:00", 220);
INSERT INTO Voo VALUES (9,  17, "2016-04-25 22:48", "2016-04-25 23:26", 200);
INSERT INTO Voo VALUES (10, 15, "2016-04-25 18:18", "2016-04-25 21:25", 210);
INSERT INTO Voo VALUES (11, 14, "2016-04-25 20:01", "2016-04-25 22:20", 217);
INSERT INTO Voo VALUES (12,  1, "2016-04-25 19:10", "2016-04-25 21:01", 200);
INSERT INTO Voo VALUES (13,  5, "2016-04-25 18:40", "2016-04-25 19:45", 200);
INSERT INTO Voo VALUES (14,  2, "2016-04-25 19:18", "2016-04-25 20:36", 150);
INSERT INTO Voo VALUES (15,  6, "2016-04-25 18:57", "2016-04-25 21:04", 190);
INSERT INTO Voo VALUES (16,  7, "2016-04-25 18:40", "2016-04-25 19:45", 200);
INSERT INTO Voo VALUES (17,  9, "2016-04-25 17:06", "2016-04-25 21:21", 203);
INSERT INTO Voo VALUES (18,  8, "2016-04-25 19:20", "2016-04-25 20:55", 196);



INSERT INTO Localidade VALUES (0, 0, "Soito");
INSERT INTO CodigoPostal VALUES (0, 0, "6320-631");
INSERT INTO Comprador VALUES (0, "Joao Silva", 4485088038682341, 271123321, 0);
INSERT INTO Reserva VALUES (0, NULL, 0);
INSERT INTO ReservaVoo VALUES (0, 0);


INSERT INTO Passageiro VALUES (0, "Joao Silva", 12312312, "1980-01-01", "390,
Estrada Municipal 538-1", 0);
INSERT INTO PassageiroNacionalidade VALUES (0, 0);
INSERT INTO ReservaPassageiro VALUES (0, 0);


INSERT INTO Detalhes VALUES (0, 0, "34A", "2 207 365 3958 3309 0", 0);
INSERT INTO Mala VALUES (0, 0, 0, 22, 40, 20, 10);
INSERT INTO Animal VALUES (0, 0, 0, 5);
INSERT INTO Animal VALUES (1, 0, 0, 7);
INSERT INTO DetalhesRefeicao VALUES (1, 0, 0);


INSERT INTO Localidade VALUES (1, 0, "Macieira de Sarnes");
INSERT INTO CodigoPostal VALUES (1, 1, "3700-714");
INSERT INTO Comprador VALUES (1, "Ricardo Rodrigues", 4539675847475440, 234555123,
0);
INSERT INTO Reserva VALUES (1, NULL, 1);
INSERT INTO ReservaVoo VALUES (0, 1);
```

```
INSERT INTO ReservaVoo VALUES (1, 1);


INSERT INTO Passageiro VALUES (1, "Ricardo Rodrigues", 32132132, "1970-06-23", "47,
Travessa 1o de Dezembro", 1);
INSERT INTO Passageiro VALUES (2, "Sara Costa", 23123123, "1975-09-05", "47,
Travessia 1o de Dezembro", 1);
INSERT INTO Passageiro VALUES (3, "Carlos Rodrigues", 33322211, "1999-04-20", "47,
Travessia 1o de Dezembro", 1);
INSERT INTO PassageiroNacionalidade VALUES (1, 0);
INSERT INTO PassageiroNacionalidade VALUES (2, 0);
INSERT INTO PassageiroNacionalidade VALUES (3, 0);
INSERT INTO ReservaPassageiro VALUES (1, 1);
INSERT INTO ReservaPassageiro VALUES (2, 1);
INSERT INTO ReservaPassageiro VALUES (3, 1);


INSERT INTO Detalhes VALUES (0, 1, "22A", "5 321 503 2103 0569 2", 0);
INSERT INTO Mala VALUES (1, 0, 1, 24, 60, 40, 20);
INSERT INTO Animal VALUES (2, 0, 1, 4);
INSERT INTO DetalhesRefeicao VALUES (0, 0, 1);

INSERT INTO Detalhes VALUES (0, 2, "22B", "5 321 503 2103 0569 3", 0);
INSERT INTO Mala VALUES (3, 0, 2, 13, 50, 30, 25);
INSERT INTO DetalhesRefeicao VALUES (3, 0, 2);

INSERT INTO Detalhes VALUES (0, 3, "22C", "5 321 503 2103 0569 4", 0);
INSERT INTO DetalhesRefeicao VALUES (0, 0, 3);


INSERT INTO Detalhes VALUES (1, 1, "36B", "8 842 038 7309 3050 5", 0);
INSERT INTO Mala VALUES (2, 1, 1, 24, 60, 40, 20);
INSERT INTO Animal VALUES (3, 1, 1, 4);
INSERT INTO DetalhesRefeicao VALUES (0, 1, 1);

INSERT INTO Detalhes VALUES (1, 2, "36A", "8 842 038 7309 3050 4", 0);
INSERT INTO Mala VALUES (4, 1, 2, 13, 50, 30, 25);
INSERT INTO DetalhesRefeicao VALUES (3, 1, 2);

INSERT INTO Detalhes VALUES (1, 3, "2A", "8 842 038 7309 3050 3", 1);
INSERT INTO DetalhesRefeicao VALUES (0, 1, 3);


INSERT INTO Localidade VALUES (2, 0, "PORTELA");
INSERT INTO Localidade VALUES (3, 0, "SACAVEM");
INSERT INTO CodigoPostal VALUES (2, 2, "2685-181");
INSERT INTO CodigoPostal VALUES (3, 3, "2685-894");
INSERT INTO Comprador VALUES (2, "Antonio Amorim", 4556024191973791, 210456456, 0);
INSERT INTO Reserva VALUES (2, NULL, 2);
INSERT INTO ReservaVoo VALUES (5, 2);
INSERT INTO ReservaVoo VALUES (6, 2);


INSERT INTO Passageiro VALUES (4, "Antonio Amorim", 88899911, "1967-06-07", "234,
Impasse I", 3);
INSERT INTO PassageiroNacionalidade VALUES (4, 0);
INSERT INTO Passageiro VALUES (5, "Armando Vieira", 44477744, "1954-03-20", "832,
Beco I", 2);
INSERT INTO PassageiroNacionalidade VALUES (5, 0);
INSERT INTO ReservaPassageiro VALUES (4, 2);
```

```
INSERT INTO ReservaPassageiro VALUES (5, 2);


INSERT INTO Detalhes VALUES (5, 4, "1A", "5 843 792 9732 2983 4", 1);
INSERT INTO Mala VALUES (5, 5, 4, 30, 40, 30, 20);
INSERT INTO DetalhesRefeicao VALUES (2, 5, 4);

INSERT INTO Detalhes VALUES (5, 5, "2A", "5 843 792 9732 2983 5", 1);
INSERT INTO Mala VALUES (6, 5, 5, 31, 40, 30, 20);
INSERT INTO DetalhesRefeicao VALUES (2, 5, 5);


INSERT INTO Detalhes VALUES (6, 4, "1A", "8 230 832 0482 3043 1", 1);
INSERT INTO Mala VALUES (7, 6, 4, 30, 40, 30, 20);
INSERT INTO DetalhesRefeicao VALUES (2, 6, 4);

INSERT INTO Detalhes VALUES (6, 5, "2A", "8 230 832 0482 3043 2", 1);
INSERT INTO Mala VALUES (8, 6, 5, 31, 40, 30, 20);
INSERT INTO DetalhesRefeicao VALUES (2, 6, 5);
```

Queries

```
-- Trajetos exitentes entre duas cidades com um maximo de 4 escalas entre elas
SELECT
        Origem.cidade AS Origem,
        Destino.cidade AS Destino,
        nEscalas,
        Origem.cidade AS Partida,
        Flights.Trajeto1,
        Escala1.cidade AS Escala1,
        Flights.Trajeto2,
        Escala2.cidade AS Escala2,
        Flights.Trajeto3,
        Escala3.cidade AS Escala3,
        Flights.Trajeto4,
        Escala4.cidade AS Escala4
FROM
        (
                SELECT
                        Origem,
                        Destino,
                        nEscalas,
                        Escala1,
                        Trajeto1,
                        Escala2,
                        Trajeto2,
                        Escala3,
                        Trajeto3,
                        Escala4,
                        Trajeto4,
                        min(distancia)
                FROM (
                        SELECT
                                L1.idAeroportoOrigem AS Origem,
                                L1.idAeroportoChegada AS Destino,
                                1 AS nEscalas,
                                L1.codigoTrajeto AS Trajeto1,
                                L1.idAeroportoChegada AS Escala1,
                                NULL AS Trajeto2,
                                NULL AS Escala2,
                                NULL AS Trajeto3,
                                NULL AS Escala3,
                                NULL AS Trajeto4,
                                NULL AS Escala4,
                                L1.distancia AS distancia
                        FROM Trajeto AS L1

                        UNION

                        SELECT
                                L1.idAeroportoOrigem AS Origem,
                                L2.idAeroportoChegada AS Destino,
                                2 AS nEscalas,
                                L1.codigoTrajeto AS Trajeto1,
                                L1.idAeroportoChegada AS Escala1,
                                L2.codigoTrajeto AS Trajeto2,
                                L2.idAeroportoChegada AS Escala2,
                                NULL AS Trajeto3,
                                NULL AS Escala3,
                                NULL AS Trajeto4,
                                NULL AS Escala4,
                                L1.distancia + L2.distancia AS distancia
                        FROM
                                Trajeto AS L1
                                INNER JOIN Trajeto AS L2
                                        ON (L1.idAeroportoChegada == L2.idAeroportoOrigem)

                        UNION

                        SELECT
                                L1.idAeroportoOrigem AS Origem,
                                L3.idAeroportoChegada AS Destino,
                                3 AS nEscalas,
                                L1.codigoTrajeto AS Trajeto1,
```

```
                                L1.idAeroportoChegada AS Escala1,
                                L2.codigoTrajeto AS Trajeto2,
                                L2.idAeroportoChegada AS Escala2,
                                L3.codigoTrajeto AS Trajeto3,
                                L3.idAeroportoChegada AS Escala3,
                                NULL AS Trajeto4,
                                NULL AS Escala4,
                                L1.distancia + L2.distancia + L3.distancia AS distancia
                        FROM
                                Trajeto AS L1
                                INNER JOIN Trajeto AS L2
                                        ON (L1.idAeroportoChegada == L2.idAeroportoOrigem)
                                INNER JOIN Trajeto AS L3
                                        ON (L2.idAeroportoChegada == L3.idAeroportoOrigem)

                        UNION

                        SELECT
                                L1.idAeroportoOrigem AS Origem,
                                L4.idAeroportoChegada AS Destino,
                                4 AS nEscalas,
                                L1.codigoTrajeto AS Trajeto1,
                                L1.idAeroportoChegada AS Escala1,
                                L2.codigoTrajeto AS Trajeto2,
                                L2.idAeroportoChegada AS Escala2,
                                L3.codigoTrajeto AS Trajeto3,
                                L3.idAeroportoChegada AS Escala3,
                                L4.codigoTrajeto AS Trajeto4,
                                L4.idAeroportoChegada AS Escala4,
                                L1.distancia + L2.distancia + L3.distancia + L4.distancia AS distancia
                        FROM
                                Trajeto AS L1
                                INNER JOIN Trajeto AS L2
                                        ON (L1.idAeroportoChegada == L2.idAeroportoOrigem)
                                INNER JOIN Trajeto AS L3
                                        ON (L2.idAeroportoChegada == L3.idAeroportoOrigem)
                                INNER JOIN Trajeto AS L4
                                        ON (L3.idAeroportoChegada == L4.idAeroportoOrigem)
                )
                WHERE distancia AND Origem != Destino
                GROUP BY Origem, Destino
                -- ORDER BY Escala4, Escala3, Escala2, Escala1
        ) AS Flights
        LEFT OUTER JOIN (SELECT cidade, idAeroporto FROM Aeroporto) AS Origem
                ON (Flights.Origem == Origem.idAeroporto)
        LEFT OUTER JOIN (SELECT cidade, idAeroporto FROM Aeroporto) AS Destino
                ON (Flights.Destino == Destino.idAeroporto)
        LEFT OUTER JOIN (SELECT cidade, idAeroporto FROM Aeroporto) AS Escala1
                ON (Flights.Escala1 == Escala1.idAeroporto)
        LEFT OUTER JOIN (SELECT cidade, idAeroporto FROM Aeroporto) AS Escala2
                ON (Flights.Escala2 == Escala2.idAeroporto)
        LEFT OUTER JOIN (SELECT cidade, idAeroporto FROM Aeroporto) AS Escala3
                ON (Flights.Escala3 == Escala3.idAeroporto)
        LEFT OUTER JOIN (SELECT cidade, idAeroporto FROM Aeroporto) AS Escala4
                ON (Flights.Escala4 == Escala4.idAeroporto)
;


-- Informação de cada Voo de cada Passageiro
SELECT
        Passageiro.nome AS Nome,
        Reserva.idReserva AS Reserva,
        Reserva.preco AS preco,
        Trajeto.codigoTrajeto AS Trajeto,
        Voo.dataPartida AS dataPartida,
        AeroportoOrigem.codigoIATA AS O_IATA,
        AeroportoOrigem.nome AS AeroportoOrigem,
        AeroportoOrigem.cidade AS CidadeOrigem,
        PaisOrigem.nome AS PaisOrigem,
        Voo.dataChegada AS dataChegada,
        AeroportoChegada.codigoIATA AS D_IATA,
        AeroportoChegada.nome AS AeroportoDestino,
        AeroportoChegada.cidade AS CidadeDestino,
        PaisChegada.nome AS PaisChegada,
        Detalhes.boardingPass AS Bilhete,
        Detalhes.lugar AS Lugar,
        Classe.nome AS Classe,
```

```
        count(Mala.idMala) AS Malas,
        count(Animal.idAnimal) AS Animais
        FROM
                (SELECT idPassageiro, nome FROM Passageiro) AS Passageiro
                INNER JOIN ReservaPassageiro
                        ON Passageiro.idPassageiro == ReservaPassageiro.idPassageiro
                INNER JOIN (SELECT idReserva, preco FROM Reserva) AS Reserva
                        ON ReservaPassageiro.idReserva == Reserva.idReserva
                INNER JOIN ReservaVoo
                        ON ReservaPassageiro.idReserva == ReservaVoo.idReserva
                INNER JOIN (SELECT idVoo, idTrajeto, dataPartida, dataChegada FROM Voo) AS Voo
                        ON ReservaVoo.idVoo == Voo.idVoo
                INNER JOIN (SELECT idTrajeto, codigoTrajeto, idAeroportoOrigem, idAeroportoChegada FROM
Trajeto) AS Trajeto
                        ON Voo.idTrajeto == Trajeto.idTrajeto
                INNER JOIN Aeroporto AS AeroportoOrigem
                        ON Trajeto.idAeroportoOrigem == AeroportoOrigem.idAeroporto
                INNER JOIN Pais AS PaisOrigem
                        ON AeroportoOrigem.idPais == PaisOrigem.idPais
                INNER JOIN Aeroporto AS AeroportoChegada
                        ON Trajeto.idAeroportoChegada == AeroportoChegada.idAeroporto
                INNER JOIN Pais AS PaisChegada
                        ON AeroportoChegada.idPais == PaisChegada.idPais
                INNER JOIN Detalhes
                        ON (Passageiro.idPassageiro == Detalhes.idPassageiro AND Voo.idVoo == Detalhes.idVoo)
                INNER JOIN Classe
                        ON Detalhes.idClasse = Classe.idClasse
                LEFT OUTER JOIN (SELECT idVoo, idPassageiro, idMala FROM MALA) AS Mala
                        ON (Passageiro.idPassageiro == Mala.idPassageiro AND Voo.idVoo == Mala.idVoo)
                LEFT OUTER JOIN (SELECT idVoo, idPassageiro, idAnimal FROM Animal) AS Animal
                        ON (Passageiro.idPassageiro == Animal.idPassageiro AND Voo.idVoo == Animal.idVoo)

GROUP BY Passageiro.idPassageiro, Reserva.idReserva, Voo.idVoo
ORDER BY Passageiro.idPassageiro, Reserva.idReserva
;


-- Passageiros com mais distancia de Voo
SELECT Passageiro.idPassageiro AS id, nome, sum(distancia) AS distanciaTotal
FROM
        (SELECT idPassageiro, nome FROM Passageiro) AS Passageiro
        INNER JOIN (SELECT idPassageiro, idReserva FROM ReservaPassageiro) AS ReservaPassageiro
                ON (Passageiro.idPassageiro == ReservaPassageiro.idPassageiro)
        INNER JOIN (SELECT idReserva, idVoo FROM ReservaVoo) AS ReservaVoo
                ON (ReservaPassageiro.idReserva == ReservaVoo.idReserva)
        INNER JOIN (SELECT idVoo, idTrajeto FROM Voo) AS Voo
                ON (ReservaVoo.idVoo == Voo.idVoo)
        INNER JOin (SELECT idTrajeto, distancia FROM Trajeto) AS Trajeto
                ON (Voo.idTrajeto == Trajeto.idTrajeto)
GROUP BY Passageiro.idPassageiro
ORDER BY sum(distancia) DESC
;


-- Compradores que gastaram mais dinheiro
SELECT Comprador.idComprador AS id, nome, sum(preco) AS dinheiroGasto
FROM
        (SELECT idReserva, idComprador, preco FROM Reserva) AS Reserva
        INNER JOIN (SELECT idComprador, nome FROM Comprador) AS Comprador
                ON (Reserva.idComprador == Comprador.idComprador)
GROUP BY id
ORDER BY sum(preco) DESC
;
```

```
-- Morada para onde mandar Malas em caso de perdas
SELECT idMala, Passageiro.nome AS Proprietario, morada, codigo, Localidade.nome AS localidade, Pais.nome AS pais
FROM
        (SELECT idPassageiro, idMala FROM Mala) AS Mala
        INNER JOIN (SELECT idPassageiro, nome, morada, idCodigoPostal FROM Passageiro) AS Passageiro
                ON (Mala.idPassageiro == Passageiro.idPassageiro)
        INNER JOIN CodigoPostal
                ON (Passageiro.idCodigoPostal == CodigoPostal.idCodigoPostal)
        INNER JOIN Localidade
                ON (CodigoPostal.idLocalidade == Localidade.idLocalidade)
        INNER JOIN Pais
                ON (Localidade.idPais == Pais.idPais)
;

-- Calcula Pbase(i, d, p) = (35 + 0.08 dist(i)) * mDiaEspecial * mCheio(p) * mDias(d)
SELECT idVoo, codigoTrajeto, dataPartida, dataChegada, codigoTrajeto, distancia, ocupacaoPercentual,
        round((35 + 0.08 * distancia) * -- preço mínimo + por km
        0.4 * (4 - 3 * (1-ocupacaoPercentual)) *  -- mCheio
        (1+6.75 / diasRestantes / diasRestantes), 2) * -- mDias, assumindo que os bilhetes estão a ser
comprados no dia 15 de abril de 2016
        -- mDiaEspecial
        CASE WHEN
        strftime("%m-%d", "2016-01-01 12:00") = strftime("%m-%d", dataPartida) OR
        strftime("%m-%d", "2016-03-25 12:00") = strftime("%m-%d", dataPartida) OR
        strftime("%m-%d", "2016-03-27 12:00") = strftime("%m-%d", dataPartida) OR
        strftime("%m-%d", "2016-04-25 12:00") = strftime("%m-%d", dataPartida) OR
        strftime("%m-%d", "2016-05-01 12:00") = strftime("%m-%d", dataPartida) OR
        strftime("%m-%d", "2016-05-26 12:00") = strftime("%m-%d", dataPartida) OR
        strftime("%m-%d", "2016-06-10 12:00") = strftime("%m-%d", dataPartida) OR
        strftime("%m-%d", "2016-07-15 12:00") = strftime("%m-%d", dataPartida) OR
        strftime("%m-%d", "2016-10-05 12:00") = strftime("%m-%d", dataPartida) OR
        strftime("%m-%d", "2016-11-01 12:00") = strftime("%m-%d", dataPartida) OR
        strftime("%m-%d", "2016-12-01 12:00") = strftime("%m-%d", dataPartida) OR
        strftime("%m-%d", "2016-12-08 12:00") = strftime("%m-%d", dataPartida) OR
        strftime("%m-%d", "2016-12-25 12:00") = strftime("%m-%d", dataPartida)
        THEN
        1.5 ELSE 1.0 END
        AS Pbase
        FROM
                (SELECT Voo.*, distancia, codigoTrajeto,
                ocupados * 1.0 / lotacao AS ocupacaoPercentual,
                strftime("%s", dataPartida) / 60 / 60 / 24 - strftime("%s", "2016-04-15 12:40") / 60 / 60 / 24
AS diasRestantes
                FROM
                        (SELECT Voo.*, count(idPassageiro) AS ocupados
                         FROM
                                 Voo
                                 LEFT OUTER JOIN Detalhes
                                         ON Voo.idVoo == Detalhes.idVoo
                         GROUP BY Voo.idVoo)
                                AS Voo
                        INNER JOIN
                                (SELECT idTrajeto, codigoTrajeto, distancia FROM Trajeto) AS Trajeto
                                        ON (Voo.idTrajeto == Trajeto.idTrajeto));


-- Query para ajudar no cálculo de Pmalas(i, j)
-- Diz para um (voo, passageiro), se tiver mala,
-- se esta é standard ou não standard e retorna o peso e dimensões dessa mesma mala
SELECT MalaStandard.idVoo, MalaStandard.idPassageiro, MalaStandard.idMala,
CASE WHEN MalaStandard IS NULL THEN
        CASE WHEN MalaNaoStandard IS NULL THEN
                "Sem Mala"
        ELSE
                "Mala Nao Standard"
        END
ELSE
        "Mala Standard"
END
AS tipoMala,
MalaStandard.peso, MalaStandard.dimensaoX, MalaStandard.dimensaoY, MalaStandard.dimensaoZ
FROM
        (SELECT Voo.idVoo, Passageiro.idPassageiro, idMala, peso, dimensaoX, dimensaoY, dimensaoZ,
        CASE WHEN
        min(dimensaoX, dimensaoY, dimensaoZ) <= 20 AND
        max(dimensaoX, dimensaoY, dimensaoZ) <=50 AND
        dimensaoX + dimensaoY + dimensaoZ - min(dimensaoX, dimensaoY, dimensaoZ) - max(dimensaoX, dimensaoY,
```

```
dimensaoZ) <= 30 AND
         peso <= 23
         THEN 1 ELSE NULL END AS MalaStandard
         FROM Voo JOIN Passageiro JOIN Detalhes LEFT OUTER JOIN Mala
         ON Mala.idVoo == Voo.idVoo AND
         Mala.idPassageiro == Passageiro.idPassageiro
         WHERE
         Voo.idVoo == Detalhes.idVoo AND
         Passageiro.idPassageiro == Detalhes.idPassageiro
         GROUP BY Voo.idVoo, Passageiro.idPassageiro)
AS MalaStandard
INNER JOIN
         (SELECT Voo.idVoo, Passageiro.idPassageiro, idMala, peso, dimensaoX, dimensaoY, dimensaoZ,
         CASE WHEN
         min(dimensaoX, dimensaoY, dimensaoZ) <= 50 AND
         max(dimensaoX, dimensaoY, dimensaoZ) <= 75 AND
         dimensaoX + dimensaoY + dimensaoZ - min(dimensaoX, dimensaoY, dimensaoZ) - max(dimensaoX, dimensaoY,
dimensaoZ) <= 150 AND
         peso <= 32
         THEN 1 ELSE NULL END AS MalaNaoStandard
         FROM Voo JOIN Passageiro JOIN Detalhes LEFT OUTER JOIN Mala
         ON Mala.idVoo == Voo.idVoo AND
         Mala.idPassageiro == Passageiro.idPassageiro
         WHERE
         Voo.idVoo == Detalhes.idVoo AND
         Passageiro.idPassageiro == Detalhes.idPassageiro
         GROUP BY Voo.idVoo, Passageiro.idPassageiro)
AS MalaNaoStandard
ON MalaStandard.idVoo == MalaNaoStandard.idVoo AND
MalaStandard.idPassageiro == MalaNaoStandard.idPassageiro;
```

Conclusão

Neste projeto chegou-se à conclusão que o desenvolvimento de uma base de dados e algoritmo para calcular o preço das suas reservas não é uma tarefa simples e exige uma extensa pré-análise de modo a que as decisões tomadas sejam as mais corretas. Existem várias formas de estruturar a base de dados, que foram consideradas até chegarmos ao modelo atual. A maximização do lucro num dado voo foi um problema difícil de resolver, e pode ser facilmente melhorado com acesso à base de dados de uma companhia aérea.

Referências

International Air Transport Association. “IATA Codes”. Iata.org.
http://www.iata.org/services/Pages/codes.asp (acedido a 19 de março de 2016).

TAP Portugal. “TAP Portugal”. flytap.com.
http://www.flytap.com (acedido a 19 de março de 2016)