

## Mestrado Integrado em Engenharia Informática e Computação

TECNOLOGIAS DE BASES DE DADOS

## Tema 1

## Modelo objeto-relacional

Eleições Legislativas 1999

André Gomes Ferreira Araújo Correia - up200706629@fe.up.pt
Artur Sousa Ferreira - ei12168@fe.up.pt
José Francisco Cagigal da Silva Gomes - up201305016@fe.up.pt

## Conteúdo

1	Inti	roduçã	ıO	4
2	Mo	delaçã	o objeto-relacional	5
	2.1	Mode	lo - Caso de estudo	5
		2.1.1	Tipos definidos pelo utilizador	5
		2.1.2	Tabelas	8
		2.1.3	Herança	9
		2.1.4	Tabelas aninhadas	10
		2.1.5	Vetores	10
		2.1.6	Referências	10
		2.1.7	Métodos para comparação e ordenação	10
3	Pov	oamer	nto da base de dados	12
4	Mé	todos	relevantes para consultas SQL	16
5	Per	guntas	5	17
	5.1	Pergu	inta $a)$ — Número total de deputados eleitos por cada partido	17
		5.1.1	SQL Procedure	17
		5.1.2	Resposta	17
	5.2	Pergu	inta $b$ ) — Número de votos em cada partido, por distrito	18
		5.2.1	SQL Procedure	18
		5.2.2	Resposta	18
	5.3	Pergu	inta $c$ ) — Partido vencedor em cada concelho	20
		5.3.1	SQL Procedure	20
		5.3.2	Resposta	20
	5.4	Pergu	inta $d$ ) — Verificar se número de inscritos é igual à soma dos votantes com o	
		númei	ro de votos nulos e brancos e abstenções	22
		5.4.1	SQL Procedure	22
		5.4.2	Resposta	22
	5.5	Pergu	inta $e)$ — Para cada partido, percentagem nacional de votos e mandatos	24
		5.5.1	SQL Procedure	24
		5.5.2	Resposta	25
	5.6	Pergu	inta $f)$ — Partidos que elegeram deputados em todos os distritos	26
		5.6.1	SQL Procedure	26
		5.6.2	Resposta	26
	5.7	Pergu	inta $g1)$ — Partido mais votado num determinado distrito	27
		5.7.1	SQL Function	27

	5.7.2 Resposta	2
5.8	Pergunta $g2$ ) — Distritos em que um determinado partido venceu	28
	5.8.1 SQL Procedure	28
	5.8.2 Resposta	28
5.9	Pergunta $g3$ ) — Distritos em que um determinado partido obteve maioria absoluta	29
	5.9.1 SQL Procedure	29
	5.9.2 Resposta	29
5.10	Pergunta $g_4$ ) — Partido mais votado num determinado distrito	3
	5.10.1 SQL Procedure	3
	5.10.2 Resposta	3
5.11	Pergunta $g5)$ — Partido que obteve maioria absoluta num determinado distrito	35
	5.11.1 SQL Procedure	35
	5.11.2 Resposta	32
5.12	Pergunta $g6$ ) — Distrito com maior número de inscritos	3
	5.12.1 SQL Function	3
	5.12.2 Resposta	3
5.13	Pergunta $g$ 7) — Distrito com maior número de votantes	3
	5.13.1 SQL Function	3
	5.13.2 Resposta	3
5.14	Pergunta $g8$ ) — Distrito com maior número de abstenções	30
	5.14.1 SQL Function	36
	5.14.2 Resposta	30
5.15	Pergunta $g9$ ) — Distrito com maior número de votos brancos	3'
	5.15.1 SQL Function	3'
	5.15.2 Resposta	3'
5.16	Pergunta $g1\theta)$ — Distrito com maior número de votos nulos	38
	5.16.1 SQL Function	38
	5.16.2 Resposta	38
5.17	Pergunta $g11)$ — Distrito com maior rácio entre o número de votantes e inscritos .	39
	5.17.1 SQL Function	39
	5.17.2 Resposta	39
5.18	Pergunta $g12)$ — Distrito com maior rácio entre o número de abstenções e inscritos	40
	5.18.1 SQL Function	40
	5.18.2 Resposta	40
5.19	Pergunta $g13)$ — Distrito com maior rácio entre o número de votos brancos e votantes	4
	5.19.1 SQL Function	4
	5.19.2 Resposta	4
5.20	Pergunta $g14)$ — Distrito com maior rácio entre o número de votos nulos e votantes	4:
	5.20.1 SQL Function	4

	5.20.2 Resposta	 	 	 42
6	Conclusão			43

## 1 Introdução

Este trabalho foi desenvolvido no âmbito da unidade curricular Tecnologias de Bases de Dados do Mestrado Integrado em Engenharia Informática e Computação, e teve como principal objetivo perceber como se pode tirar partido da utilização da tecnologia *Object-Oriented* em bases de dados relacionais, nomeadamente, através da utilização de tipos de dados complexos, objetos e respetivos métodos para manipular a sua estrutura de dados, usufruindo da herança e polimorfismo, referências, vetores e tabelas aninhadas.

O trabalho foi desenvolvido em torno do tema das eleições legislativas de 1999, e teve na sua génese uma base de dados relacional contendo os resultados de tais eleições.

No decorrer deste relatório vão ser analisadas as opções tomadas pelos autores, no que à modelação da base de dados diz respeito, e algumas consultas executadas sobre essa mesma base de dados. De entre essas consultas, algumas constituem resposta às perguntas presentes no enunciado do trabalho - da a) à f) - e outras, dada a sua relevância para o tema em análise, partiram da iniciativa dos autores — da g1) à g14).

## 2 Modelação objeto-relacional

#### 2.1 Modelo - Caso de estudo

A Figura 1 ilustra o diagrama referente à modelação adotada pelos autores para o caso de estudo. A modelação adotada é descrita em pormenor na secção seguinte.

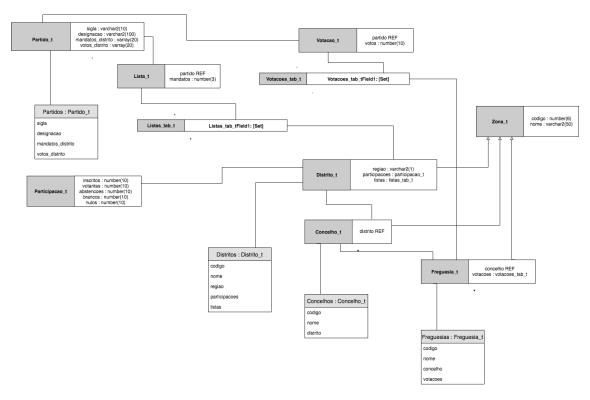


Figura 1: Diagrama do modelo OR adotado.

#### 2.1.1 Tipos definidos pelo utilizador

Relativamente ao caso em análise, foram definidos os seguintes tipos:

 partido\_mandatos — representa um array de 20 posições (correspondentes ao número de distritos), que tem por objetivo armazenar o número de mandatos obtido por cada partido, em cada um dos distritos.

```
CREATE OR REPLACE TYPE partido_mandatos AS VARRAY(20) OF NUMBER(3);
```

partido\_votos — representa um array de 20 posições (correspondentes ao número de distritos), que tem por objetivo armazenar o número de votos obtido por cada partido, em cada um dos distritos.

```
CREATE OR REPLACE TYPE partido_votos AS VARRAY(20) OF NUMBER(10);
```

• partido\_t — representa um determinado partido. Cada partido tem uma sigla, designação, um vetor do tipo mandatos\_distrito e outro do tipo votos\_distrito. Estes vetores têm como

objetivo armazenar, respetivamente, o número de mandatos e o número de votos obtido por cada partido, em cada um dos distritos. Conforme explicado em 3, ambos os vetores são preenchidos com recurso a um procedure. O tipo  $partido\_t$  tem, ainda, uma função membro que retorna a sigla (e que é utilizada para comparar e ordenar objetos do tipo  $partido\_t$ ) e duas funções membro —  $get\_total\_mandatos$  e  $get\_total\_votos$  — que retornam, respetivamente, o número total de mandatos e o número total de votos de um partido.

```
CREATE OR REPLACE TYPE partido_t AS OBJECT(
    sigla VARCHAR2(10),
    designacao VARCHAR2(100),
    mandatos_distrito partido_mandatos,
    votos_distrito partido_votos,
    map member function get_sigla return varchar2,
    member function get_total_mandatos return number,
    member function get_total_votos return number
);
```

 lista\_t — representa uma candidatura de um partido a um determinado distrito. Os atributos do tipo lista\_t são uma referência para o partido em causa e o número de mandatos obtido num distrito.

```
CREATE OR REPLACE TYPE lista_t AS OBJECT(
    partido REF partido_t,
    mandatos NUMBER(3)
);
```

 listas\_tab\_t — representa uma tabela que guarda objetos do tipo lista\_t, referido na alínea anterior.

```
CREATE OR REPLACE TYPE listas_tab_t AS TABLE OF lista_t;
```

 participacao\_t — representa informação estatística de um ato eleitoral num determinado distrito, nomeadamente, o número de inscritos, o número de cidadãos que participaram na eleição e o número de abstenções, votos brancos e nulos.

```
CREATE OR REPLACE TYPE participacao_t AS OBJECT(
    inscritos NUMBER(10),
    votantes NUMBER(10),
    abstencoes NUMBER(10),
    brancos NUMBER(10),
    nulos NUMBER(10))
);
```

 votacao\_t — representa o resultado de uma candidatura de um partido numa determinada freguesia. Os atributos do tipo votacao\_t são uma referência para o partido em causa e o número de votos obtido numa freguesia.

```
CREATE OR REPLACE TYPE votacao_t AS OBJECT(
```

```
partido REF partido_t ,
  votos NUMBER(10)
);
```

• votacoes\_tab\_t — representa uma tabela que guarda objetos do tipo votacao\_t, referido na alínea anterior.

```
CREATE OR REPLACE TYPE votacoes_tab_t AS TABLE OF votacao_t;
```

• zona\_t — representa o supertipo dos subtipos distrito\_t, concelho\_t e freguesia\_t. São atributos do supertipo zona\_t o código e o nome. O supertipo zona\_t possui, ainda, duas funções membro — get\_codigo e get\_nome — que retornam, respetivamente, o código e o nome. A função membro get\_codigo é utilizada para comparar e ordenar objetos do tipo e subtipos zona\_t.

```
CREATE OR REPLACE TYPE zona_t AS OBJECT(
    codigo NUMBER(6),
    nome VARCHAR2(50),
    map member function get_codigo return number,
    member function get_nome return varchar2
) not final;
```

• distrito\_t — representa um distrito. O tipo distrito\_t é um subtipo do supertipo zona\_t. Tem como atributos a região em que se encontra, informação estatística referente ao ato eleitoral (tipo participacao\_t) e uma tabela aninhada com o número de votos que cada partido obteve no distrito (tipo listas\_tab\_t). Possui, ainda, uma função membro get\_nome que substitui a função do supertipo zona\_t.

```
CREATE OR REPLACE TYPE distrito_t under zona_t(
    regiao VARCHAR2(1),
    participacoes participacao_t,
    listas listas_tab_t,
    overriding member function get_nome return varchar2
);
```

• concelho\_t — representa um concelho. O tipo concelho\_t é um subtipo do supertipo zona\_t.

Tem como atributo uma referência para o distrito a que pertence. Possui, ainda, uma função membro get\_nome que substitui a função do supertipo zona\_t.

```
CREATE OR REPLACE TYPE concelho_t under zona_t(
    distrito REF distrito_t,
    overriding member function get_nome return varchar2
);
```

• freguesia\_t — representa uma freguesia. O tipo freguesia\_t é um subtipo do supertipo zona\_t. Tem como atributos uma referência para o concelho a que pertence e uma tabela aninhada (tipo votacoes\_tab\_t), com informação sobre número de votos que cada partido

obteve na freguesia. Possui, ainda, uma função membro  $get\_nome$  que substitui a função do supertipo zona $\_$ t.

```
CREATE OR REPLACE TYPE freguesia_t under zona_t(
    concelho REF concelho_t,
    votacoes votacoes_tab_t,
    overriding member function get_nome return varchar2
);
```

A Tabela 1 resume todos os tipos criados.

Tipos			
freguesia_t			
$concelho\_t$			
distrito_t			
zona_t			
votacoes_tab_t			
lista_t			
partido_t			
$partido\_votos$			
partido_mandatos			
nome_distritos			
votacao_t			
participacao_t			
listas_tab_t			

Tabela 1: Lista de tipos criados

#### 2.1.2 Tabelas

Conforme se constata através da análise do script abaixo, foram criadas 4 tabelas: partidos (do tipo  $partido\_t$ ), distritos (do tipo  $distrito\_t$ ), concelhos (do tipo  $concelho\_t$ ) e freguesias (do tipo  $freguesia\_t$ .

```
CREATE TABLE partidos OF partido_t;

CREATE TABLE distritos OF distrito_t
    NESTED TABLE listas STORE AS listas_tab;

CREATE TABLE concelhos OF concelho_t;

CREATE TABLE freguesias OF freguesia_t
    NESTED TABLE votacoes STORE AS votacoes_tab;
```

#### 2.1.3 Herança

Tirando partido do facto de as tabelas distrito, concelho e freguesia terem, em comum, as colunas código e nome, foi criado o tipo zona\_t, contendo tais atributos. Este tipo funciona, assim, como supertipo para os subtipos distrito\_t, concelho\_t e freguesia\_t, que herdam aqueles dois atributos de zona\_t. O supertipo zona\_t possui, ainda, duas funções membro representadas no código abaixo.

```
CREATE OR REPLACE TYPE BODY zona_t AS

map member function get_codigo return number is

begin

return codigo;

end get_codigo;

member function get_nome return varchar2 is

begin

return nome;

end get_nome;

end;
```

Contudo, pode ver-se através da análise do segmento de código seguinte, que os tipos  $distrito\_t$ ,  $concelho\_t$  e  $freguesia\_t$  substituem a função membro  $get\_nome$ .

```
CREATE OR REPLACE TYPE BODY distrito_t AS
    overriding member function get_nome return varchar2 is
        begin
    return 'Distrito: ' || nome;
    end get_nome;
end;
```

```
CREATE OR REPLACE TYPE BODY concelho_t AS
    overriding member function get_nome return varchar2 is
    begin
    return 'Concelho: ' || nome;
    end get_nome;
end;
```

```
CREATE OR REPLACE TYPE BODY freguesia_t AS
    overriding member function get_nome return varchar2 is
    begin
    return 'Freguesia: ' || nome;
    end get_nome;
end;
```

Assim, tirando partido do polimorfismo, sempre que se chamar a função membro *get\_nome*, associada a um dos subtipos *distrito\_t*, *concelho\_t* ou *freguesia\_t*, a função executada é a referente a um dos três subtipos anteriores e não a função do supertipo *zona\_t*.

#### 2.1.4 Tabelas aninhadas

Existem duas tabelas aninhadas, estando elas representadas nos dois segmentos de código abaixo.

```
CREATE TABLE distritos OF distrito_t

NESTED TABLE listas STORE AS listas_tab;

CREATE TABLE freguesias OF freguesia_t

NESTED TABLE votacoes STORE AS votacoes_tab;
```

A tabela distritos tem uma tabela aninhada de listas, armazenando informação sobre os partidos que apresentaram candidatura no distrito e quantos mandatos conquistaram.

Por sua vez, a tabela *freguesias* possui uma tabela aninhada de votações, que guarda informação sobre o número de votos que os partidos obtiveram na freguesia.

#### 2.1.5 Vetores

Conforme referido em 2.1.1, na modelação adotada foram utilizados 2 vetores diferentes — partido\_mandatos e partido\_votos.

Relativamente ao partido\_mandatos, este vetor foi criado com o intuito de guardar o número de mandatos de um partido por distrito. Como existem 20 partidos, o tamanho do vetor é de 20, ou seja, cada posição corresponde a um distrito. A posição do vetor guarda o número de mandatos que determinado partido teve no distrito correspondente à posição. Se o partido não apresentou candidatura a um determinado distrito, a respetiva posição fica com o valor null.

À semelhança do tipo anterior, o vetor *partido\_votos* difere apenas quanto ao objetivo, que é guardar o número de votos de um partido por distrito.

#### 2.1.6 Referências

No tipo *lista\_t* é guardada uma referência para o partido a que o número de mandatos corresponde. Seguindo a mesma lógica, no tipo *votacao\_t* é referenciado o partido a quem os votos pertencem. Por sua vez, no tipo *freguesia\_t* é guardada uma referência para o concelho a que a freguesia pertence e, no tipo *concelho\_t*, é referenciado o distrito a que o mesmo pertence.

#### 2.1.7 Métodos para comparação e ordenação

Conforme referido em 2.1.1, o tipo partido\_t tem uma função map member get\_sigla como demonstra o código abaixo. Isto permite que os objetos deste tipo possam ser comparados usando o atributo sigla. Outra possibilidade é o facto de ao serem efetuadas consultas à tabela dos partidos, estas poderem ser organizadas recorrendo a esta função.

```
map member function get_sigla return varchar2 is
   begin
   return sigla;
   end get_sigla;
```

À semelhança do caso anterior, o tipo zona\_t também possui uma função map member, denominada get\_codigo. Desta forma, é possível comparar os objetos do tipo ou subtipos de zona\_t, utilizando o seu código e ordenar as pesquisas nas tabelas freguesias, concelhos e distritos recorrendo a esta função.

```
map member function get_codigo return number is
    begin
    return codigo;
    end get_codigo;
member function get_nome return varchar2 is
```

### 3 Povoamento da base de dados

Como é possível constatar através da análise do *script* em baixo, o povoamento das tabelas partidos, distritos, concelhos e freguesias foi feito tendo por base as respetivas tabelas do utilizador gtd7. Salienta-se o facto de, os INSERTS nas tabelas distritos e freguesias serem mais complexos dos que os restantes, uma vez que tais tabelas possuem mais colunas do que as respetivas tabelas do utilizador gtd7.

Com efeito, no caso da tabela distritos, a inserção na coluna participacoes é feita com recurso à criação de um objeto  $participacao\_t$ , com os atributos resultantes da tabela participacoes do utilizador gtd7. Por sua vez, a inserção na coluna listas é feita com recurso ao CAST(MULTISET(...)), por forma a permitir que o resultado da interrogação, à tabela listas do utilizador gtd7, seja utilizado para a inserção de uma tabela aninhada numa linha.

Relativamente à tabela freguesias, a inserção na coluna votacoes é, igualmente, feita com recurso ao CAST(MULTISET(...)), por forma a permitir que o resultado da interrogação, à tabela votacoes do utilizador gtd7, seja utilizado para a inserção de uma tabela aninhada numa linha.

```
INSERTS
  Partidos -
INSERT INTO partidos (sigla, designacao)
SELECT *
FROM gtd7.partidos;
 - Distritos --
INSERT INTO distritos (codigo, nome, regiao, participacoes, listas)
SELECT d.codigo, d.nome, d.regiao, participacao_t(p.inscritos, p.votantes, p.
    abstencees, p. brancos, p. nulos), cast (multiset (SELECT (SELECT REF(part) from
    partidos part where sigla = l.partido), l.mandatos
                                              FROM gtd7.listas l WHERE l.distrito =
    d.codigo) as listas_tab_t)
FROM gtd7. distritos d
JOIN gtd7. participacoes p
ON d.codigo = p.distrito;
 - Concelhos --
```

Por forma a facilitar a inserção de dados nos dois vetores que fazem parte do tipo partido\_t — partido\_mandatos e partido\_votos — foram desenvolvidos pelos autores dois procedures — fill\_partido\_mandatos e fill\_partido\_votos.

Relativamente ao procedure fill\_partido\_mandatos, conforme é possível verificar através na análise do script em baixo, são efetuados dois ciclos — um que percorre todos os partidos e outro que, para cada partido, percorre os distritos — por forma a obter os mandatos conseguidos por cada partido, em cada um dos distritos, e armazenar essa informação no respetivo vetor partido mandatos.

```
- FILL PARTIDO MANDATOS -
CREATE OR REPLACE PROCEDURE fill_partido_mandatos
 _{\rm IS}
                                    mandatos array \quad partido\_mandatos := partido\_mandatos (NULL, NULL, NUL
                                 NULL, 
                                    distritoposarray SIMPLE_INTEGER := 1;
                                    mandatos VARCHAR2(3) := '';
BEGIN
                                 FOR P IN (SELECT sigla FROM partidos)
                                 LOOP
                                                                      FOR D IN (SELECT D. codigo, L. mandatos FROM distritos D, TABLE(D. listas) L
                                WHERE L. partido. sigla = P. sigla)
                                                                     LOOP
                                                                                                        IF(D. codigo <= 18)
                                                                                                        THEN
                                                                                                                                               distritoposarray := D.codigo;
```

Em relação ao procedure fill\_partido\_votos, como é possível constatar através na análise do script em baixo, são também efetuados dois ciclos — um que percorre todos os partidos e outro que, para cada partido, percorre os distritos — por forma a obter a soma de todos os votos conseguidos por cada partido, em cada uma das freguesias pertencentes a cada um dos distritos, e armazenar essa informação no respetivo vetor partido\_votos.

```
- FILL_PARTIDO_VOTOS -
CREATE OR REPLACE PROCEDURE fill_partido_votos
  _{\rm IS}
                                      votosarray \quad partido\_votos := partido\_votos (NULL, NULL, N
                                  NULL, 
                                      distritoposarray SIMPLE_INTEGER := 1;
                                      votos VARCHAR2(10) := ', ';
  BEGIN
                                   FOR P IN (SELECT sigla FROM partidos)
                                  LOOP
                                                                        FOR D IN (SELECT codigo FROM distritos)
                                                                       LOOP
                                                                                                            BEGIN
                                                                                                                                               SELECT SUM(x.votos) INTO votos
                                                                                                                                              FROM freguesias F, TABLE(F. votacoes) x
                                                                                                                                               WHERE x.partido.sigla = P.sigla AND F.concelho.codigo IN
```

```
(SELECT codigo
                                                                    FROM concelhos C
                                                                    WHERE C. distrito.codigo = D. codigo)
                                                                    GROUP BY x.partido.sigla;
                                                                    EXCEPTION
                                                                            WHEN no_data_found THEN
                                                                                     votos := NULL;
                                                   END;
                                                   IF (D. codigo <= 18)
                                                   THEN
                                                                     {\tt distritoposarray} \; := \; {\tt D.} \; {\tt codigo} \; ;
                                                   ELSIF(D.codigo = 30)
                                                   THEN
                                                                     distritoposarray := 19;
                                                   ELSE
                                                                     distritoposarray := 20;
                                                   END IF;
                                                   IF NOT (votos IS NULL)
                                                                     votosarray(distritoposarray) := to_number(votos);
                                                   END IF;
                                  END LOOP;
                                  UPDATE partidos
                                  SET votos_distrito = votosarray
                                  WHERE partidos.sigla = P.sigla;
                                   votosarray := partido\_votos (NULL, NULL, NULL,
                NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL);
                                   distritoposarray := 1;
                END LOOP;
END;
```

## 4 Métodos relevantes para consultas SQL

Conforme é possível constatar através da análise do *script* em baixo, ao tipo *partido\_t*, foram adicionados dois métodos — *get\_total\_mandatos* e *get\_total\_votos*.

O primeiro percorre o vetor *mandatos\_distrito*, somando todos os mandatos obtidos pelo partido em cada distrito, e devolve o valor correspondente à soma.

O segundo percorre o vetor *votos\_distrito*, somando todos os votos obtidos pelo partido em cada distrito, e devolve o valor correspondente à soma.

O método  $get\_total\_mandatos$  é utilizado para responder à pergunta 5.1 — número total de deputados eleitos por cada partido ( $procedure~GET\_TOTAL\_MANDATOS\_PARTIDOS$ ) — e, juntamente com o método  $get\_total\_votos$ , são ambos utilizados para responder à pergunta 5.5 — para cada partido, percentagem nacional de votos e mandatos (procedure  $GET\_TOTAL\_MANDATOS\_VOTOS\_PARTS$ ).

```
CREATE OR REPLACE TYPE BODY partido_t AS
    map member function get_sigla return varchar2 is
        begin
        return sigla;
        end get_sigla;
    member function get_total_mandatos return number is
        total number(3) := 0;
        begin
        for \ m \ in \ 1...mandatos\_distrito.count
        if not (mandatos_distrito(m) is null)
        then
        total := total + mandatos_distrito(m);
        end if;
        end loop;
        return total;
        end get_total_mandatos;
    member function get_total_votos return number is
        total number(7) := 0;
        begin
        for m in 1.. votos_distrito.count
        loop
        if not(votos_distrito(m) is null)
        total := total + votos_distrito(m);
        end if;
        end loop;
        return total;
        end get_total_votos;
end;
```

## 5 Perguntas

### 5.1 Pergunta a) — Número total de deputados eleitos por cada partido

#### 5.1.1 SQL Procedure

```
CREATE OR REPLACE PROCEDURE GET_TOTAL_MANDATOS_PARTIDOS

IS

rowpartido partidos%ROWIYPE;
CURSOR c1 IS SELECT value(p) FROM partidos p;

BEGIN

OPEN c1;

LOOP

FETCH c1 INTO rowpartido;
EXIT WHEN c1%NOIFOUND;

dbms_output.put_line(rowpartido.get_sigla() || ': ' || rowpartido.get_total_mandatos());
END LOOP;

CLOSE c1;
END;
```

#### 5.1.2 Resposta

A Figura 2 ilustra o resultado da execução do *procedure* anterior, representando o número total de deputados eleitos por cada partido.

```
Connecting to the database TBDA.
PS: 112
PSN: 0
PPM: 0
PPDPSD: 80
POUS: 0
PDA: 0
PH: 0
PCTPMRPP: 0
PCTPMRPP: 17
CDSPP: 15
BE: 2
MPT: 0
Process exited.
Disconnecting from the database TBDA.
```

Figura 2: Resultados da execução da consulta da pergunta a).

### 5.2 Pergunta b) — Número de votos em cada partido, por distrito

#### 5.2.1 SQL Procedure

```
- GET_TOTAL_VOTOS_PARTIDOS_DISTR -
CREATE OR REPLACE PROCEDURE GET_TOTAL_VOTOS_PARTIDOS_DISTR
    distrito VARCHAR2(50) := ';
    cod SIMPLE\_INTEGER := 1;
BEGIN
   FOR I IN 1..20
   LOOP
        IF(I = 19)
        THEN
            cod := 30;
        ELSIF(I = 20)
        THEN
            cod := 40;
        END IF;
        SELECT nome INTO distrito
        FROM distritos
        WHERE codigo = cod;
        cod := cod + 1;
        dbms_output.put_line('Distrito: ' || distrito);
        FOR P IN (SELECT * FROM partidos)
        LOOP
            IF NOT(P.votos_distrito(I) IS NULL)
                dbms\_output.put\_line('--'||P.\,sigla||': '|| \ P.\,votos\_distrito(I));
            END IF;
        END LOOP;
    END LOOP;
END;
```

#### 5.2.2 Resposta

A Figura 3 ilustra um excerto do resultado da execução do *procedure* anterior, representando, para cada distrito, o número de votos que cada partido obteve. De salientar que, em cada distrito, apenas são apresentados os resultados dos partidos que apresentaram candidatura nesse mesmo distrito.

```
Connecting to the database TBDA.
Distrito: Aveiro
--PS: 145575
--PSN: 660
--PPM: 1148
--PPDPSD : 138686
--PH: 968
--PCTPMRPP: 1511
--PCPPEV : 12797
--CDSPP : 49183
--BE: 4676
--MPT: 847
Distrito: Beja
--PS: 39728
--PSN: 207
--PPM: 393
--PPDPSD: 12308
--PCTPMRPP: 1664
--PCPPEV : 24077
--CDSPP: 3315
--BE: 1316
--MPT: 279
Distrito: Braga
--PS: 195602
--PSN: 1758
--PPM: 1434
--PPDPSD: 162433
--POUS: 830
--PH: 873
--PCTPMRPP: 2947
--PCPPEV: 23821
--CDSPP: 39027
--BE: 5164
--MPT: 973
Distrito: Bragança
--PS: 32588
--PSN: 230
--PPDPSD: 36841
--PCTPMRPP: 530
--PCPPEV: 2141
--CDSPP: 7079
--BE: 679
--MPT: 296
```

Figura 3: Resultados da execução da consulta da pergunta b).

## 5.3 Pergunta c) — Partido vencedor em cada concelho

#### 5.3.1 SQL Procedure

```
CREATE OR REPLACE PROCEDURE GET_PARTIDO_VENCEDOR_CONCELHO

IS

sigla partidos.sigla%TYPE;

BEGIN

FOR C IN (SELECT nome, codigo FROM concelhos)

LOOP

SELECT x.partido.sigla INTO sigla

FROM freguesias F, TABLE(F.votacoes) x

WHERE F.concelho.codigo = C.codigo

GROUP BY x.partido.sigla

ORDER BY SUM(x.votos) DESC

FETCH FIRST ROW ONLY;

dbms_output.put_line(C.nome||': '|| sigla);

END LOOP;

END;
```

#### 5.3.2 Resposta

A Figura 4 ilustra um excerto do resultado da execução do *procedure* anterior, representando o partido vencedor em cada concelho.

Connecting to the database TBDA. Mirandela: PPDPSD Mogadouro: PPDPSD Vila Flor: PPDPSD Vimioso: PPDPSD Vinhais: PS Belmonte: PS Castelo Branco: PS Covilhã: PS Fundão: PS Oleiros: PPDPSD Penamacor: PS Proença-a-Nova: PPDPSD Sertã: PPDPSD Vila Velha de Rodão: PS Arganil: PPDPSD Odivelas: PS Trofa: PS Vizela: PS Ansião: PPDPSD Batalha: PPDPSD Bombarral: PS Caldas da Rainha: PPDPSD Castanheira de Pera: PS Figueiró dos Vinhos: PPDPSD Leiria: PPDPSD Marinha Grande: PS Nazaré: PS Óbidos: PS Peniche: PS Pombal: PPDPSD Porto de Mós: PPDPSD

Figura 4: Resultados da execução da consulta da pergunta c).

# 5.4 Pergunta d) — Verificar se número de inscritos é igual à soma dos votantes com o número de votos nulos e brancos e abstenções

#### 5.4.1 SQL Procedure

```
CHECK_NUMERO_VOTANTES —
CREATE OR REPLACE PROCEDURE check_numero_votantes
    numvotos NUMBER(10);
    numinscritos distritos.participacoes.inscritos%TYPE;
    numabsbrancosnulos distritos.participacoes.inscritos%TYPE;
BEGIN
   FOR D IN (SELECT codigo, nome FROM distritos)
   LOOP
       SELECT SUM(x.votos) INTO numvotos
       FROM freguesias F, TABLE(F.votacoes) x
       WHERE F. concelho. codigo IN
        (SELECT codigo
       FROM concelhos C
       WHERE C. distrito.codigo = D. codigo);
       SELECT dist.participacoes.inscritos, (dist.participacoes.abstencoes + dist.
    participacoes.brancos + dist.participacoes.nulos)
        INTO numinscritos, numabsbrancosnulos
        FROM distritos dist
       WHERE dist.codigo = D.codigo;
        IF NOT(numvotos + numabsbrancosnulos = numinscritos)
            dbms_output.put_line('No distrito de ' || D.nome || chr(10) || ' n o
    se verifica a restri o de o n mero de inscritos ser igual ao n mero de
    votantes mais o n mero de absten es mais o n mero de votos brancos e nulos
    ');
       END IF;
   END LOOP;
END;
```

#### 5.4.2 Resposta

A Figura 5 ilustra o resultado da execução do *procedure* anterior, representando os distritos em que não é cumprida a restrição de o número de inscritos ter de ser igual à soma do número de votantes, com o número de votos brancos e nulos e com as abstenções. Conforme é possível constatar através da análise da Figura 5, apenas nos Açores não é cumprida tal restrição.

Connecting to the database TBDA.

No distrito de Açores
não se verifica a restrição de o número de inscritos ser igual ao número de votantes mais o número de abstenções mais o número de votos brancos e nulos
Process exited.
Disconnecting from the database TBDA.

Figura 5: Resultados da execução da consulta da pergunta d).

## 5.5 Pergunta e) — Para cada partido, percentagem nacional de votos e mandatos

#### 5.5.1 SQL Procedure

```
----- GET_TOTAL_MANDATOS_VOTOS_PARTIDOS ---
CREATE OR REPLACE PROCEDURE GET_TOTAL_MANDATOS_VOTOS_PARTS
    rowpartido partidos%ROWTYPE;
    CURSOR c1 IS SELECT value(p) FROM partidos p;
    votos number (5,2) := 0;
    mandatos number (5,2) := 0;
    totalVotos number(7) := 0;
    totalMandatos number(3) := 0;
BEGIN
    OPEN c1;
    LOOP
        FETCH c1 INTO rowpartido;
        EXIT WHEN c1%NOTFOUND;
             totalVotos := totalVotos + rowpartido.get_total_votos();
             total Mandatos \; := \; total Mandatos \; + \; rowpartido \, . \, get\_total\_mandatos \, () \; ;
    END LOOP;
    CLOSE c1;
    OPEN c1;
    LOOP
        FETCH c1 INTO rowpartido;
        EXIT WHEN c1%NOTFOUND;
         votos := rowpartido.get_total_votos()/totalVotos*100;
         mandatos := rowpartido.get_total_mandatos()/totalMandatos*100;
        dbms\_output.put\_line(rowpartido.get\_sigla() \ || \ chr(10) \ || \ 'Percentagem
    votos: ' || TO_CHAR(votos, '90D00') || '%' || chr(10) || 'Percentagem mandatos:
    ^{,} || TO_CHAR(mandatos, ^{,}90\,\mathrm{D}00\,^{,}) || ^{,}\% || \mathrm{chr}\left(10\right) || ^{,}
                            ----·');
    END LOOP;
    CLOSE c1;
END;
```

#### 5.5.2 Resposta

A Figura 6 ilustra o resultado da execução do *procedure* anterior, representando as percentagens nacionais de votos e mandatos de cada partido.

Connecting to the database TBDA. Percentagem votos: 44,89% Percentagem mandatos: 49,56% **PSN** Percentagem votos: 0,22% Percentagem mandatos: 0,00% Percentagem votos: 0,31% Percentagem mandatos: 0,00% **PPDPSD** Percentagem votos: 32,98% Percentagem mandatos: 35,40% **POUS** Percentagem votos: 0,08% Percentagem mandatos: 0,00% Percentagem votos: 0,01% Percentagem mandatos: 0,00% Percentagem votos: 0,14% Percentagem mandatos: 0,00% **PCTPMRPP** Percentagem votos: 0,75% Percentagem mandatos: 0,00% Percentagem votos: 9,20% Percentagem mandatos: 7,52% **CDSPP** Percentagem votos: 8,55% Percentagem mandatos: 6,64% Percentagem votos: 2,51% Percentagem mandatos: 0,88% **MPT** Percentagem votos: 0,37% Percentagem mandatos: 0,00% Process exited. Disconnecting from the database TBDA.

Figura 6: Resultados da execução da consulta da pergunta e).

## 5.6 Pergunta f) — Partidos que elegeram deputados em todos os distritos

#### 5.6.1 SQL Procedure

```
—— GET_PARTIDO_PLENO_DISTRTITO —
CREATE OR REPLACE PROCEDURE get_partido_pleno_distrtito
    sigla partidos.sigla%TYPE;
    pleno BOOLEAN := TRUE;
    FOR P IN (SELECT sigla, mandatos_distrito FROM partidos)
   LOOP
        FOR I IN 1..P. mandatos_distrito.COUNT
        LOOP
            IF (P.mandatos\_distrito(I) IS NULL OR P.mandatos\_distrito(I) = 0)
            THEN
                pleno := FALSE;
                EXIT;
            END IF;
        END LOOP;
        IF(pleno = TRUE)
        THEN
            dbms_output.put_line('O' | | P.sigla | | ' elegeu deputados em todos os
    distritos');
        END IF;
        pleno := TRUE;
   END LOOP;
END;
```

#### 5.6.2 Resposta

A Figura 7 ilustra o resultado da execução do *procedure* anterior, representando os partidos que conseguiram eleger deputados em todos os distritos. Conforme é possível constatar através da análise da Figura 7, apenas o PS conseguiu tal feito.

```
Connecting to the database TBDA.

O PS elegeu deputados em todos os distritos
Process exited.

Disconnecting from the database TBDA.
```

Figura 7: Resultados da execução da consulta da pergunta f).

### 5.7 Pergunta g1) — Partido mais votado num determinado distrito

#### 5.7.1 SQL Function

```
- GET_PARTIDO_MAIS_VOTADO -
 \hbox{\tt CREATE OR REPLACE FUNCTION GET\_PARTIDO\_MAIS\_VOTADO(codigoDistrito \ distritos.codigo\%) } \\
RETURN partidos.sigla%TYPE
    Sigla partidos.sigla%TYPE;
BEGIN
    SELECT x.partido.sigla INTO Sigla
    FROM freguesias F, TABLE(F. votacoes) x
    WHERE F. concelho. codigo IN
        (SELECT codigo
        FROM concelhos C
        WHERE C. distrito.codigo = codigoDistrito)
    GROUP BY x.partido.sigla
    ORDER BY SUM(x.votos) DESC
    FETCH FIRST ROW ONLY;
RETURN Sigla;
END;
```

#### 5.7.2 Resposta

A Figura 8 ilustra o resultado da execução da função anterior, representando o partido que obteve maior número de votos num determinado distrito. O resultado presente na Figura 8, diz respeito ao distrito com o código '1' (Aveiro).

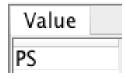


Figura 8: Resultados da execução da consulta da pergunta g1).

### 5.8 Pergunta g2) — Distritos em que um determinado partido venceu

#### 5.8.1 SQL Procedure

```
CREATE OR REPLACE PROCEDURE get_distritos_vencedor(siglapartido partidos.sigla%TYPE

)

IS

sigla partidos.sigla%TYPE := '';

ret_distritos VARCHAR2(1000) := '';

BEGIN

FOR D IN (SELECT codigo, nome FROM distritos)

LOOP

sigla := get_partido_mais_votado(D.codigo);

IF sigla = siglapartido

THEN

ret_distritos := ret_distritos || D.nome || CHR(10);

END IF;

END LOOP;

dbms_output.put_line(ret_distritos);

END;
```

#### 5.8.2 Resposta

A Figura 9 ilustra o resultado da execução do *procedure* anterior, representando os distritos em que um determinado partido venceu as eleições. O resultado presente na Figura 9, diz respeito ao partido 'PPDPSD'.

Connecting to the database TBDA.
Bragança
Leiria
Vila Real
Viseu
Madeira
Process exited.
Disconnecting from the database TBDA.

Figura 9: Resultados da execução da consulta da pergunta g2).

## 5.9 Pergunta g3) — Distritos em que um determinado partido obteve maioria absoluta

#### 5.9.1 SQL Procedure

```
— GET_DISTRITOS_MAIORIA —
CREATE OR REPLACE PROCEDURE get_distritos_maioria(siglapartido partidos.sigla%TYPE)
    sigla partidos.sigla%TYPE := '';
    ret_distritos VARCHAR2(1000) := '';
    mandatos_proprio simple_integer := 0;
    mandatos_outros simple_integer := 0;
BEGIN
   FOR D IN (SELECT codigo, nome FROM distritos)
   LOOP
        SELECT L. mandatos INTO mandatos_proprio
        FROM distritos dist, TABLE(dist.listas) L
        WHERE dist.codigo = D.codigo AND L.partido.sigla = siglapartido;
        SELECT SUM(L.mandatos) INTO mandatos_outros
        FROM distritos dist, TABLE(dist.listas) L
        WHERE dist.codigo = D.codigo AND L.partido.sigla \Leftrightarrow siglapartido;
        IF mandatos_proprio > mandatos_outros
        THEN
            ret_distritos := ret_distritos || D.nome || CHR(10);
        END IF;
   END LOOP;
    dbms_output.put_line(ret_distritos);
END;
```

#### 5.9.2 Resposta

A Figura 10 ilustra o resultado da execução do *procedure* anterior, representando os distritos em que um determinado partido obteve maioria absoluta. O resultado presente na Figura 10, diz respeito ao partido 'PS'.

Connecting to the database TBDA.

Beja

Castelo Branco

Coimbra

Faro

Portalegre

Porto

Açores

Process exited.

Disconnecting from the database TBDA.

Figura 10: Resultados da execução da consulta da pergunta g3).

## 5.10 Pergunta $g_4$ ) — Partido mais votado num determinado distrito

#### 5.10.1 SQL Procedure

```
- GET_PARTIDO_MAIS_VOTADO -
 {\tt CREATE~OR~REPLACE~PROCEDURE~get\_partido\_mais\_votado\_distr(nomedistrito~distritos~.} \\
    nome%TYPE)
IS
    sigla partidos.sigla%TYPE;
BEGIN
    SELECT x.partido.sigla INTO sigla
    FROM freguesias F, TABLE(F. votacoes) x
    WHERE F. concelho. codigo IN
        (SELECT codigo
        FROM concelhos C
        WHERE C. distrito.nome = nomedistrito)
    GROUP BY x.partido.sigla
    ORDER BY SUM(x.votos) DESC
    FETCH FIRST ROW ONLY;
    dbms_output.put_line(sigla);
END;
```

#### 5.10.2 Resposta

A Figura 11 ilustra o resultado da execução do *procedure* anterior, representando o partido que obteve maior número de votos num determinado distrito. O resultado presente na Figura 11, diz respeito ao distrito de 'Vila Real'.

Connecting to the database TBDA.

PPDPSD

Process exited.

Disconnecting from the database TBDA.

Figura 11: Resultados da execução da consulta da pergunta g4).

## 5.11 Pergunta g5) — Partido que obteve maioria absoluta num determinado distrito

#### 5.11.1 SQL Procedure

```
— GET_PARTIDO_MAIORIA —
CREATE OR REPLACE PROCEDURE get_partido_maioria_distr(nomedistrito distritos.nome%
    TYPE)
_{\rm IS}
    nome distritos.nome%TYPE := ',';
    nomePartido VARCHAR2(10) := ';
    mandatos\_max SIMPLE\_INTEGER := 0;
    mandatos_outros SIMPLE_INTEGER := 0;
BEGIN
    SELECT L. partido. sigla, L. mandatos
    INTO nomePartido, mandatos_max
    FROM distritos dist, TABLE(dist.listas) L
    WHERE dist.nome = nomedistrito
    ORDER BY L. mandatos DESC
    FETCH FIRST ROW ONLY;
    SELECT SUM(L. mandatos) INTO mandatos_outros
    FROM distritos dist, TABLE(dist.listas) L
    WHERE dist.nome = nomedistrito AND L.partido.sigla \Leftrightarrow nomePartido;
    IF mandatos_max > mandatos_outros
        dbms_output.put_line(nomePartido);
    ELSE
        dbms_output.put_line('Nenhum partido obteve maioria absoluta no distrito de
     ' || nomedistrito);
    END IF;
END;
```

#### 5.11.2 Resposta

As Figuras 12 e 13 ilustram o resultado da execução do *procedure* anterior, representando o partido que obteve maioria absoluta num determinado distrito. O resultado presente na Figura 12, diz respeito ao distrito de 'Braga', em que nenhum partido conseguiu tal feito.

Connecting to the database TBDA.

Nenhum partido obteve maioria absoluta no distrito de Braga

Process exited.

Disconnecting from the database TBDA.

Figura 12: Resultados da execução da consulta da pergunta g5).

O resultado presente na Figura 13, diz respeito ao distrito de 'Porto', em o que o 'PS' obteve maioria absoluta.

Connecting to the database TBDA.

PS

Process exited.

Disconnecting from the database TBDA.

Figura 13: Resultados da execução da consulta da pergunta g5).

## 5.12 Pergunta g6) — Distrito com maior número de inscritos

#### 5.12.1 SQL Function

```
CREATE OR REPLACE FUNCTION GET_DISTRITO_MAIS_INSCRITOS
RETURN distritos.nome%TYPE
IS
Nome distritos.nome%TYPE;
BEGIN

SELECT D.nome INTO Nome
FROM distritos D
ORDER BY D.participacoes.inscritos DESC
FETCH FIRST ROW ONLY;

RETURN Nome;
```

#### 5.12.2 Resposta

A Figura 14 ilustra o resultado da execução da função anterior, representando o distrito com maior número de inscritos. Conforme é possível constatar através da análise da Figura 14, tal distrito é o distrito de 'Lisboa'.

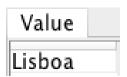


Figura 14: Resultados da execução da consulta da pergunta g6).

## 5.13 Pergunta g7) — Distrito com maior número de votantes

#### 5.13.1 SQL Function

```
CREATE OR REPLACE FUNCTION GET_DISTRITO_MAIS_VOTANTES
RETURN distritos.nome%TYPE
IS
Nome distritos.nome%TYPE;
BEGIN

SELECT D.nome INTO Nome
FROM distritos D
ORDER BY D.participacoes.votantes DESC
FETCH FIRST ROW ONLY;

RETURN Nome;
END;
```

#### 5.13.2 Resposta

A Figura 15 ilustra o resultado da execução da função anterior, representando o distrito com maior número de votantes. Conforme é possível constatar através da análise da Figura 15, tal distrito é o distrito de 'Lisboa'.

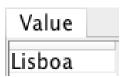


Figura 15: Resultados da execução da consulta da pergunta g7).

## 5.14 Pergunta g8) — Distrito com maior número de abstenções

#### 5.14.1 SQL Function

```
GET_DISTRITO_MAIS_ABSTENCOES

CREATE OR REPLACE FUNCTION GET_DISTRITO_MAIS_ABSTENCOES

RETURN distritos.nome%TYPE

IS

Nome distritos.nome%TYPE;

BEGIN

SELECT D.nome INTO Nome
FROM distritos D

ORDER BY D.participacoes.abstencoes DESC
FETCH FIRST ROW ONLY;

RETURN Nome;

END;
```

#### 5.14.2 Resposta

A Figura 16 ilustra o resultado da execução da função anterior, representando o distrito com maior número de abstenções. Conforme é possível constatar através da análise da Figura 16, tal distrito é o distrito de 'Lisboa'.

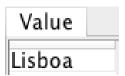


Figura 16: Resultados da execução da consulta da pergunta g8).

## 5.15 Pergunta g9) — Distrito com maior número de votos brancos

#### 5.15.1 SQL Function

```
CREATE OR REPLACE FUNCTION GET_DISTRITO_MAIS_BRANCOS
RETURN distritos.nome%TYPE
IS
Nome distritos.nome%TYPE;
BEGIN

SELECT D.nome INTO Nome
FROM distritos D
ORDER BY D.participacoes.brancos DESC
FETCH FIRST ROW ONLY;

RETURN Nome;
```

#### 5.15.2 Resposta

A Figura 17 ilustra o resultado da execução da função anterior, representando o distrito com maior número de votos brancos. Conforme é possível constatar através da análise da Figura 17, tal distrito é o distrito de 'Lisboa'.

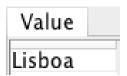


Figura 17: Resultados da execução da consulta da pergunta g9).

## 5.16 Pergunta g10) — Distrito com maior número de votos nulos

#### 5.16.1 SQL Function

```
CREATE OR REPLACE FUNCTION GET_DISTRITO_MAIS_NULOS
RETURN distritos.nome%TYPE
IS
Nome distritos.nome%TYPE;
BEGIN

SELECT D.nome INTO Nome
FROM distritos D
ORDER BY D.participacoes.nulos DESC
FETCH FIRST ROW ONLY;

RETURN Nome;
```

#### 5.16.2 Resposta

A Figura 18 ilustra o resultado da execução da função anterior, representando o distrito com maior número de votos nulos. Conforme é possível constatar através da análise da Figura 18, tal distrito é o distrito de 'Lisboa'.

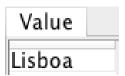


Figura 18: Resultados da execução da consulta da pergunta g10).

# 5.17 Pergunta g11) — Distrito com maior rácio entre o número de votantes e inscritos

#### 5.17.1 SQL Function

```
CREATE OR REPLACE FUNCTION GET_DISTR_MAIOR_RACIO_VOT_INSC
RETURN distritos.nome%IYPE
IS
Nome distritos.nome%IYPE;
BEGIN

SELECT D.nome INTO NOME
FROM distritos D
ORDER BY D.participacoes.votantes/D.participacoes.inscritos DESC
FETCH FIRST ROW ONLY;

RETURN Nome;
```

#### 5.17.2 Resposta

A Figura 19 ilustra o resultado da execução da função anterior, representando o distrito com maior rácio entre o número de votantes e o número de inscritos. Conforme é possível constatar através da análise da Figura 19, tal distrito é o distrito de 'Braga'.



Figura 19: Resultados da execução da consulta da pergunta g11).

# 5.18 Pergunta g12) — Distrito com maior rácio entre o número de abstenções e inscritos

#### 5.18.1 SQL Function

#### 5.18.2 Resposta

A Figura 20 ilustra o resultado da execução da função anterior, representando o distrito com maior rácio entre o número de abstenções e o número de inscritos. Conforme é possível constatar através da análise da Figura 20, são os Açores que possuem o maior valor para tal rácio.

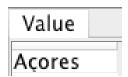


Figura 20: Resultados da execução da consulta da pergunta g12).

# 5.19 Pergunta g13) — Distrito com maior rácio entre o número de votos brancos e votantes

#### 5.19.1 SQL Function

```
CREATE OR REPLACE FUNCTION GET_DISTR_MAIOR_RACIO_BRNC_VOT
RETURN distritos.nome%TYPE
IS
Nome distritos.nome%TYPE;
BEGIN

SELECT D.nome INTO NOME
FROM distritos D
ORDER BY D. participacoes.brancos/D. participacoes.votantes DESC
FETCH FIRST ROW ONLY;

RETURN Nome;
```

#### 5.19.2 Resposta

A Figura 21 ilustra o resultado da execução da função anterior, representando o distrito com maior rácio entre o número de votos brancos e o número de votantes. Conforme é possível constatar através da análise da Figura 21, tal distrito é o distrito de 'Faro'.

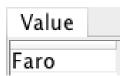


Figura 21: Resultados da execução da consulta da pergunta g13).

# 5.20 Pergunta g14) — Distrito com maior rácio entre o número de votos nulos e votantes

#### 5.20.1 SQL Function

```
CREATE OR REPLACE FUNCTION GET_DISTR_MAIOR_RACIO_NUL_VOT
RETURN distritos.nome%TYPE

IS

Nome distritos.nome%TYPE;

BEGIN

SELECT D.nome INTO NOME
FROM distritos D

ORDER BY D.participacoes.nulos/D.participacoes.votantes DESC
FETCH FIRST ROW ONLY;

RETURN Nome;
```

#### 5.20.2 Resposta

A Figura 22 ilustra o resultado da execução da função anterior, representando o distrito com maior rácio entre o número de votos nulos e o número de votantes. Conforme é possível constatar através da análise da Figura 22, tal distrito é o distrito da 'Guarda'.

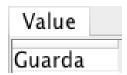


Figura 22: Resultados da execução da consulta da pergunta g14).

## 6 Conclusão

Após a realização do presente trabalho, é possível concluir que a combinação da tecnologia Object-Oriented com bases de dados relacionais é de extrema importância e utilidade, pois permite, não apenas, a definição e utilização de tipos complexos, mas sobretudo armazenar todos os dados referentes a um determinado objeto apenas numa tabela. Este facto traduz-se na enorme vantagem, do ponto de vista da eficiência computacional, de não ser necessário o recurso à operação de junção de tabelas (a operação mais custosa em SQL).

Não obstante, é de salientar que a utilização deste paradigma traz consigo o problema da necessidade de armazenamento redundante da informação, implicando, por isso, um maior número de verificações por forma a garantir que não se verificam inconsistências na informação armazenada.

## Referências

[G.18] David G. Object relational assignment - parliament elections of october, 10th 1999. Database Technology - Integrated Master in Informatics Engineering, Faculty of Engineering of the University of Porto, 2018.