

Otimização de Interrogações

Tecnologias de Bases de Dados

Catarina Felgueiras

up202403074

José Francisco Veiga

up202108753

Mestrado em Engenharia Informática e Computação

Faculdade de Engenharia da Universidade do Porto



8 de abril de 2025

Resumo

Este relatório apresenta uma análise detalhada da otimização de interrogações SQL sobre uma base de dados que contém informações relativas às eleições legislativas de 1999 em Portugal, focando-se na avaliação do impacto de diferentes estratégias de indexação e restrições de integridade na eficiência de consultas. Para tal, foram criados três ambientes distintos, nos quais foram analisadas interrogações que abrangem operações de seleção, junção, agregação e negação, além de uma consulta universal e a comparação entre índices do tipo árvore-B e bitmap. Cada consulta foi executada em todos os ambientes, avaliando o plano de execução e o tempo de resposta, permitindo demonstrar o impacto da indexação e da estruturação das consultas na performance e extrair conclusões sobre a otimização de bases de dados relacionais.

Palavras-chave: Base de dados relacionais; SQL; Otimização; Consulta; Interrogações

Índice

| | | |
|----------|--|-----------|
| 1 | Introdução | 3 |
| 2 | Descrição detalhada dos Ambientes | 3 |
| 2.1 | Ambiente X | 3 |
| 2.2 | Ambiente Y | 4 |
| 2.3 | Ambiente Z | 5 |
| 3 | Queries | 8 |
| 4 | Resultados | 17 |
| 5 | Conclusão | 37 |

1 Introdução

Para garantir a escalabilidade de sistemas de base de dados, é crucial a otimização de interrogações SQL crucial, sobretudo quando estes sistemas lidam com grandes volumes de dados e consultas bastante complexas. O foco deste trabalho é a análise e comparação de diferentes estratégias de otimização de consultas, avaliando o impacto que a utilização de índices e a aplicação de restrições de integridade, por exemplo, têm na eficiência de interrogações realizadas sobre uma base de dados real. O conjunto de dados utilizados são informações relativas às eleições para a Assembleia da República, ocorridas a 10 de Outubro de 1999, abrangendo a hierarquia administrativa do país – que se organiza em distritos, concelhos e freguesias – bem como os resultados eleitorais, a distribuição de mandatos e os votos registados por cada partido.

Para conduzir esta análise, foram estabelecidos três ambientes de teste distintos:

- Ambiente X: Base de dados sem índices nem restrições de integridade.
- Ambiente Y: Base de dados com restrições de integridade standard, que incluem chaves primárias e estrangeiras.
- Ambiente Z: Base de dados com as restrições de integridade standard e índices adicionais, cujo impacto na otimização das queries é avaliado.

Como objetivo deste trabalho, são criadas e analisadas queries que abrangem operações de seleção, junção, agregação e negação, além de consultas universais e comparações entre diferentes estratégias de indexação (como índices do tipo árvore-B e bitmap). Cada consulta é executada nos três ambientes, permitindo uma avaliação detalhada dos planos de execução e do tempo de resposta, de forma a evidenciar a influência das técnicas de otimização adotadas na performance global do sistema.

2 Descrição detalhada dos Ambientes

2.1 Ambiente X

No enunciado do trabalho, o Ambiente X foi definido como um cenário experimental sem índices nem restrições de integridade. Ao criar as tabelas `xdistritos`, `xconcelhos`, `xfreguesias`, `xpartidos`, `xvotacoes`, `xlistas` e `xparticipacoes` a partir das tabelas originais presentes no user `GTD7`, estamos a criar uma cópia original dos dados, sem a presença de quaisquer mecanismos que possam otimizar as queries, como índices ou chaves.

Esta abordagem tem dois objetivos:

- **Avaliar o desempenho básico:** Sem quaisquer otimizações, o tempo de resposta e o plano de execução das consultas refletem o desempenho “bruto” da base de dados.
- **Estabelecer uma linha de base para comparações:** Ao comparar os resultados do ambiente X com os ambientes Y (com restrições de integridade) e Z (com restrições de integridade e índices adicionais), é possível mensurar o impacto real das otimizações na execução das consultas.

```
1 CREATE TABLE xdistritos AS SELECT * FROM GTD7.districts;  
2 CREATE TABLE xconcelhos AS SELECT * FROM GTD7.concelhos;  
3 CREATE TABLE xfreguesias AS SELECT * FROM GTD7.freguesias;  
4 CREATE TABLE xpartidos AS SELECT * FROM GTD7.partidos;
```

```

5 CREATE TABLE xvotacoes AS SELECT * FROM GTD7.votacoes;
6 CREATE TABLE xlistas AS SELECT * FROM GTD7.listas;
7 CREATE TABLE xparticipacoes AS SELECT * FROM GTD7.participacoes;

```

Listing 1: Criação de tabelas no ambiente X

2.2 Ambiente Y

No enunciado do trabalho, o Ambiente Y é definido como um cenário experimental em que as cópias dos dados do user GTD7 são enriquecidas com restrições de integridade, ou seja, com chaves primárias e estrangeiras. Esta configuração procurar adicionar regras de integridade referencial e garantir a consistência dos dados, permitindo ainda avaliar o impacto dessas restrições na execução das interrogações, comparativamente com o Ambiente X.

```

1 CREATE TABLE ydistritos AS SELECT * FROM GTD7.distritos;
2 CREATE TABLE yconcelhos AS SELECT * FROM GTD7.concelhos;
3 CREATE TABLE yfreguesias AS SELECT * FROM GTD7.freguesias;
4 CREATE TABLE ypartidos AS SELECT * FROM GTD7.partidos;
5 CREATE TABLE yvotacoes AS SELECT * FROM GTD7.votacoes;
6 CREATE TABLE ylistas AS SELECT * FROM GTD7.listas;
7 CREATE TABLE yparticipacoes AS SELECT * FROM GTD7.participacoes;
8
9 ALTER TABLE "YDISTRITOS" ADD CONSTRAINT YDISTRITOS_CODIGO_PK PRIMARY KEY(
10     "CODIGO");
11 ALTER TABLE "YCONCELHOS" ADD CONSTRAINT YCONCELHOS_CODIGO_PK PRIMARY KEY(
12     "CODIGO");
13 ALTER TABLE "YFREGUESIAS" ADD CONSTRAINT YFREGUESIAS_CODIGO_PK PRIMARY
14     KEY("CODIGO");
15 ALTER TABLE "YPARTIDOS" ADD CONSTRAINT YPARTIDOS_SIGLA_PK PRIMARY KEY("
16     SIGLA");
17 ALTER TABLE "YVOTACOES" ADD CONSTRAINT YVOTACOES_PARTIDO_FREGUESIA_PK
18     PRIMARY KEY("PARTIDO", "FREGUESIA");
19 ALTER TABLE "YLISTAS" ADD CONSTRAINT YLISTAS_DISTRITO_PARTIDO_PK PRIMARY
20     KEY("DISTRITO", "PARTIDO");
21 ALTER TABLE "YPARTICIPACOES" ADD CONSTRAINT YPARTICIPACOES_DISTRITO_PK
22     PRIMARY KEY("DISTRITO");
23
24 ALTER TABLE "YCONCELHOS" ADD CONSTRAINT YCONCELHOS_DISTRITO_FK FOREIGN
25     KEY("DISTRITO") REFERENCES "YDISTRITOS"("CODIGO");
26
27 ALTER TABLE "YFREGUESIAS" ADD CONSTRAINT YFREGUESIAS_CONCELHO_FK FOREIGN
28     KEY("CONCELHO") REFERENCES "YCONCELHOS"("CODIGO");
29
30 ALTER TABLE "YVOTACOES" ADD CONSTRAINT YVOTACOES_PARTIDO_FK FOREIGN KEY("
31     PARTIDO") REFERENCES "YPARTIDOS"("SIGLA");
32 ALTER TABLE "YVOTACOES" ADD CONSTRAINT YVOTACOES_FREGUESIA_FK FOREIGN KEY
33     ("FREGUESIA") REFERENCES "YFREGUESIAS"("CODIGO");
34
35 ALTER TABLE "YLISTAS" ADD CONSTRAINT YLISTAS_DISTRITO_FK FOREIGN KEY("
36     DISTRITO") REFERENCES "YDISTRITOS"("CODIGO");
37 ALTER TABLE "YLISTAS" ADD CONSTRAINT YLISTAS_PARTIDO_FK FOREIGN KEY("
38     PARTIDO") REFERENCES "YPARTIDOS"("SIGLA");
39

```

```
27 ALTER TABLE "YPARTICIPACOES" ADD CONSTRAINT YPARTICIPACOES_DISTRITO_FK
    FOREIGN KEY("DISTRITO") REFERENCES "YDISTRITOS"("CODIGO");
```

Listing 2: Criação de tabelas e restrições de integridade no Ambiente Y

- **Chaves primárias:** Garantem a unicidade dos registos em cada tabela (por exemplo, o campo "CODIGO" em YDISTRITOS, YCONCELHOS e YFREGUESIAS; "SIGLA" em YPARTIDOS; combinações de PARTIDO e FREGUESIA em YVOTACOES, entre outros.).
- **Chaves estrangeiras:** Asseguram a integridade referencial entre as tabelas, fazendo com que os valores presentes nos campos que fazem referência a outras tabelas correspondam a registos existentes nas tabelas referenciadas (por exemplo, o campo "DISTRITO" em YCONCELHOS refere-se ao campo "CODIGO" em YDISTRITOS).

Com esta abordagem, o Ambiente Y possibilita uma avaliação mais realista do desempenho das interrogações, uma vez que reflete as condições comuns de um sistema de bases de dados em produção, onde as restrições de integridade garantem a manutenção da consistência dos dados.

2.3 Ambiente Z

No enunciado do trabalho, o Ambiente Z é definido como um cenário experimental em que os dados originais são enriquecidos não só com restrições de integridade como acontece no Ambiente Y, mas também com índices adicionais criados por nós. Ou seja, além de garantir a integridade referencial, neste ambiente introduz-se um conjunto de índices com o objetivo otimizar a execução das interrogações SQL.

O principal objetivo desta configuração é comparar a eficiência das queries em função da existência de índices apropriados, algo que não é possível observar nos outros Ambientes.

```
1 CREATE TABLE zdistritos AS SELECT * FROM GTD7.districts;
2 CREATE TABLE zconcelhos AS SELECT * FROM GTD7.concelhos;
3 CREATE TABLE zfreguesias AS SELECT * FROM GTD7.freguesias;
4 CREATE TABLE zpartidos AS SELECT * FROM GTD7.partidos;
5 CREATE TABLE zvotacoes AS SELECT * FROM GTD7.votacoes;
6 CREATE TABLE zlistas AS SELECT * FROM GTD7.listas;
7 CREATE TABLE zparticipacoes AS SELECT * FROM GTD7.participacoes;
8
9 ALTER TABLE "ZDISTRITOS" ADD CONSTRAINT ZDISTRITOS_CODIGO_PK PRIMARY KEY(
    "CODIGO");
10 ALTER TABLE "ZCONCELHOS" ADD CONSTRAINT ZCONCELHOS_CODIGO_PK PRIMARY KEY(
    "CODIGO");
11 ALTER TABLE "ZFREGUESIAS" ADD CONSTRAINT ZFREGUESIAS_CODIGO_PK PRIMARY
    KEY("CODIGO");
12 ALTER TABLE "ZPARTIDOS" ADD CONSTRAINT ZPARTIDOS_SIGLA_PK PRIMARY KEY("
    SIGLA");
13 ALTER TABLE "ZVOTACOES" ADD CONSTRAINT ZVOTACOES_PARTIDO_FREGUESIA_PK
    PRIMARY KEY("PARTIDO", "FREGUESIA");
14 ALTER TABLE "ZLISTAS" ADD CONSTRAINT ZLISTAS_DISTRITO_PARTIDO_PK PRIMARY
    KEY("DISTRITO", "PARTIDO");
15 ALTER TABLE "ZPARTICIPACOES" ADD CONSTRAINT ZPARTICIPACOES_DISTRITO_PK
    PRIMARY KEY("DISTRITO");
16
17 ALTER TABLE "ZCONCELHOS" ADD CONSTRAINT ZCONCELHOS_DISTRITO_FK FOREIGN
    KEY("DISTRITO") REFERENCES "ZDISTRITOS"("CODIGO");
```

```

18
19 ALTER TABLE "ZFREGUESIAS" ADD CONSTRAINT ZFREGUESIAS_CONCELHO_FK FOREIGN
    KEY("CONCELHO") REFERENCES "ZCONCELHOS"("CODIGO");
20
21 ALTER TABLE "ZVOTACOES" ADD CONSTRAINT ZVOTACOES_PARTIDO_FK FOREIGN KEY("
    PARTIDO") REFERENCES "ZPARTIDOS"("SIGLA");
22 ALTER TABLE "ZVOTACOES" ADD CONSTRAINT ZVOTACOES_FREGUESIA_FK FOREIGN KEY
    ("FREGUESIA") REFERENCES "ZFREGUESIAS"("CODIGO");
23
24 ALTER TABLE "ZLISTAS" ADD CONSTRAINT ZLISTAS_DISTRITO_FK FOREIGN KEY("
    DISTRITO") REFERENCES "ZDISTRITOS"("CODIGO");
25 ALTER TABLE "ZLISTAS" ADD CONSTRAINT ZLISTAS_PARTIDO_FK FOREIGN KEY("
    PARTIDO") REFERENCES "ZPARTIDOS"("SIGLA");
26
27 ALTER TABLE "ZPARTICIPACOES" ADD CONSTRAINT ZPARTICIPACOES_DISTRITO_FK
    FOREIGN KEY("DISTRITO") REFERENCES "ZDISTRITOS"("CODIGO");
28
29 CREATE INDEX IDX_FREGUESIAS_CONCELHO ON ZFREGUESIAS(CONCELHO);
30 CREATE INDEX IDX_CONCELHOS_DISTRITO ON ZCONCELHOS(DISTRITO);
31 CREATE INDEX IDX_CONCELHOS_NOME ON ZCONCELHOS(NOME);
32 CREATE INDEX IDX_DISTRITOS_NOME ON ZDISTRITOS(NOME);
33
34 CREATE INDEX IDX_VOTACOES_PARTIDO ON ZVOTACOES(PARTIDO);
35 CREATE INDEX IDX_VOTACOES_PARTIDO_VOTOS ON ZVOTACOES(PARTIDO, VOTOS);
36 CREATE INDEX IDX_VOTACOES_FREGUESIA_VOTOS ON ZVOTACOES(FREGUESIA, VOTOS
    DESC);
37 CREATE INDEX IDX_VOTACOES_PARTIDO_FREGUESIA ON ZVOTACOES(PARTIDO,
    FREGUESIA, VOTOS);
38 CREATE INDEX IDX_VOTACOES_FREGUESIA ON ZVOTACOES(FREGUESIA);

```

Listing 3: Criação de tabelas e restrições de integridade e índices no Ambiente Z

Explicação dos índices

Os índices numa base de dados são estruturas cujo objetivo é melhorar o desempenho das consultas, permitindo que o sistema acesse os dados de modo muito mais rápido e mais eficiente. Em vez de percorrer todas as linhas de uma tabela (o chamado full table scan, que será mencionada mais à frente durante a análise de resultados), é utilizado o índice como uma espécie de “guia”, localizando rapidamente as linhas que satisfazem as condições de pesquisa definidas na query. Assim, quando se faz uma filtragem por determinadas colunas, ou quando se estabelecem junções entre diferentes tabelas, o uso de índices bem escolhidos reduz o tempo de resposta. Além disso, a criação de índices adequados, como em colunas onde se faz frequentemente pesquisas por igualdade ou por intervalos, não sobrecarrega tanto o sistema e torna o acesso a grandes quantidades de dados mais eficiente. Em consequência, as operações de leitura e recuperação de informação tornam-se mais rápidas, o que influencia positivamente o desempenho.

IDX_FREGUESIAS_CONCELHO em ZFREGUESIAS (CONCELHO):

Este índice facilita a seleção de freguesias pertencentes a um concelho específico. Útil para a questão 1.a), onde se filtram as freguesias de um dado concelho, e para as junções entre freguesias e concelhos. Este índice permite localizar diretamente os blocos de disco onde estão armazenadas as freguesias associadas a um determinado concelho. Sem ele, seria necessário percorrer toda a tabela de freguesias (leitura sequencial), verificando linha a linha.

IDX_CONCELHOS_DISTRITO em ZCONCELHOS (DISTRITO):

Acelera a procura dos concelhos que pertencem a um distrito, otimizando a junção entre essas tabelas. Este índice é essencial para queries que relacionam concelhos e distritos, como nas questões que envolvem agregação ou filtragem por distrito. Ao existir este índice, o motor de execução da consulta consegue localizar rapidamente todos os concelhos que partilham o mesmo código de distrito. A estrutura indexada reduz o número de páginas de disco lidas e evita fazer uma leitura total da tabela, tal como acontece no índice anteriormente apresentado.

IDX_CONCELHOS_NOME em ZCONCELHOS (NOME):

Melhora o desempenho de buscas por nome de concelho (por exemplo, ao procurar “Azambuja” na questão 1a). Este índice facilita a filtragem textual por nome de concelho. A procura por valores textuais como “Azambuja” pode ser lenta sem um índice, especialmente se a tabela tiver muitos registos. Este índice permite aplicar pesquisas rápidas por nome, pois os dados estão ordenados e organizados para acesso direto.

IDX_DISTritos_NOME em ZDISTritos (NOME):

Este índice permite filtrar rapidamente os distritos com base no seu nome, como por exemplo “Lisboa” ou “Porto”, sendo essencial em diversas consultas onde se pretende isolar ou cruzar dados de um distrito específico, como nas questões 1b), 1c), 3 e 4.

Quando não existe um índice sobre esta coluna, o sistema precisa de percorrer toda a tabela ZDISTritos para encontrar o nome pretendido, o que pode ser ineficiente se a tabela tiver muitos registos. Com o índice, a procura textual é feita diretamente sobre uma estrutura ordenada, permitindo identificar rapidamente o distrito com o nome desejado, com um número mínimo de leituras e comparações.

IDX_VOTACOES_PARTIDO em ZVOTACOES (PARTIDO):

Este índice acelera as consultas que filtram as votações por partido, como é o caso das questões 2a), 2b) e 5, onde se pretende calcular o número de votos obtidos pelo PS ou PSD.

Sem este índice, o sistema teria de fazer uma busca completa à tabela ZVOTACOES para localizar as linhas associadas a um partido específico. Com o índice, conseguem-se localizar diretamente os blocos onde estão armazenadas as votações desse partido, reduzindo a carga de leitura. Isto é especialmente vantajoso em operações de agregação, onde o número de votos por partido é calculado de forma eficiente, mesmo em grandes volumes de dados.

IDX_VOTACOES_PARTIDO_VOTOS em ZVOTACOES (PARTIDO, VOTOS):

Permite aceder diretamente ao valor dos votos quando se filtra pelo partido, sem a necessidade de ler outras colunas. Especialmente útil em queries de agregação, como a soma de votos por partido (questão 2a) e na comparação de estratégias (questão 5).

IDX_VOTACOES_FREGUESIA_VOTOS em ZVOTACOES (FREGUESIA, VOTOS DESC):

Este índice ordena os registos de votos por freguesia e, dentro de cada uma, por votos de forma decrescente. Otimiza a extração rápida do registo com o maior número de votos por freguesia, usado na questão 2c (para identificar o partido com maior votação) ou na questão 4. Evita ordenações manuais na consulta e acesso a registos irrelevantes, diminuindo a carga de leitura.

IDX_VOTACOES_PARTIDO_FREGUESIA em ZVOTACOES (PARTIDO, FREGUESIA, VOTOS):

Este índice composto atende a consultas que filtram por partido e, em seguida, por freguesia, tendo ainda o valor dos votos incluído na estrutura. Ideal para as agregações e junções em queries que envolvem o total de votos de um partido ou comparações entre partidos, como nas questões 1c e 2a.

IDX_VOTACOES_FREGUESIA em ZVOTACOES (FREGUESIA):

Este índice é essencial para melhorar o desempenho das junções entre a tabela ZVOTACOES e ZFREGUESIAS, onde as votações de cada freguesia são relacionadas com os seus metadados.

Aparece frequentemente em interrogações como as das questões 1c), 2c) e 4, onde se analisam as votações a nível de freguesia. Com este índice, o acesso às votações de uma freguesia concreta é direto, sem necessidade de verificar todos os registos da tabela ZVOTACOES. Este índice é ainda útil em subconsultas que envolvem filtros por FREGUESIA, como aquelas que procuram o partido com mais votos numa dada freguesia.

3 Queries

Questão 1 - Seleção e junção

a) Quais os códigos e nomes de freguesias do concelho 1103? E do concelho “Azambuja”?

```

1 SELECT codigo, nome
2 FROM freguesias
3 WHERE freguesias.concelho = 1103;
4
5 SELECT freguesias.codigo, freguesias.nome
6 FROM freguesias, concelhos
7 WHERE freguesias.concelho = concelhos.codigo
8       AND concelhos.nome = 'Azambuja';

```

Listing 4: Consultas SQL para obter freguesias do concelho 1103 e de “Azambuja”

O resultado desta query é o seguinte:

| | CODIGO | NOME |
|---|--------|------------------------|
| 1 | 110301 | Alcoentre |
| 2 | 110302 | Aveiras de Baixo |
| 3 | 110303 | Aveiras de Cima |
| 4 | 110304 | Azambuja |
| 5 | 110305 | Manique do Intendente |
| 6 | 110306 | Vale do Paraíso |
| 7 | 110307 | Vila Nova da Rainha |
| 8 | 110308 | Vila Nova de São Pedro |
| 9 | 110309 | Maçussa |

Figura 1: Resultado Query 1.a).

b) Indique as siglas e designações dos partidos e o respetivo número de mandatos obtidos no distrito de Lisboa.

```

1 SELECT partidos.sigla, partidos.designacao, listas.mandatos
2 FROM partidos, listas, distritos
3 WHERE distritos.nome = 'Lisboa'
4       AND distritos.codigo = listas.distrito
5       AND partidos.sigla = listas.partido
6 ORDER BY listas.mandatos DESC;

```

Listing 5: Mandatos por partido no distrito de Lisboa

O resultado desta query é o seguinte:

| | SIGLA | DESIGNACAO | MANDATOS |
|----|----------|---|----------|
| 1 | PS | Partido Socialista | 23 |
| 2 | PPDPSD | Partido Social Democrata | 14 |
| 3 | PCPPEV | Partido Comunista Português | 6 |
| 4 | CDSPP | Partido Popular | 4 |
| 5 | BE | Bloco de Esquerda | 2 |
| 6 | MPT | Movimento Partido da Terra | 0 |
| 7 | PSN | Partido Solidariedade Nacional | 0 |
| 8 | PH | Partido Humanista | 0 |
| 9 | POUS | Partido Operário de Unidade Socialista | 0 |
| 10 | PPM | Partido Popular Monárquico | 0 |
| 11 | PCTPMRPP | Partido Comunista dos Trabalhadores Portugueses | 0 |

Figura 2: Resultado Query 1.b).

c) Indique o número de votos obtido pelo BE nas freguesias do distrito de Lisboa.

```

1 SELECT SUM(votacoes.votos) AS votos_be
2 FROM votacoes, freguesias, concelhos, distritos
3 WHERE distritos.nome = 'Lisboa'
4     AND distritos.codigo = concelhos.distrito
5     AND concelhos.codigo = freguesias.concelho
6     AND freguesias.codigo = votacoes.freguesia
7     AND votacoes.partido = 'BE';

```

Listing 6: Total de votos do BE no distrito de Lisboa

O resultado desta query é o seguinte:

| VOTOS_BE |
|----------|
| 1 55113 |

Figura 3: Resultado Query 1.c).

Questão 2 - Agregação

a) Quantos votos teve o 'PS' a nível nacional?

```

1 SELECT SUM(votos) AS votos_ps
2 FROM votacoes
3 WHERE partido = 'PS';

```

Listing 7: Total de votos do PS

O resultado desta query é o seguinte:

| VOTOS_PS |
|-----------|
| 1 2359939 |

Figura 4: Resultado Query 2.a).

b) Quantos votos teve cada partido, em cada distrito?

```

1 SELECT distritos.nome, partidos.designacao, SUM(votacoes.votos) AS votos
2 FROM distritos, concelhos, freguesias, votacoes, partidos
3 WHERE distritos.codigo = concelhos.distrito
4       AND concelhos.codigo = freguesias.concelho
5       AND freguesias.codigo = votacoes.freguesia
6       AND votacoes.partido = partidos.sigla
7 GROUP BY distritos.nome, partidos.designacao
8 ORDER BY distritos.nome, votos DESC;

```

Listing 8: Total de votos por partido em cada distrito

O resultado desta query é o seguinte:

| | NOME | DESIGNACAO | VOTOS |
|----|--------|---|--------|
| 1 | Açores | Partido Socialista | 49947 |
| 2 | Açores | Partido Social Democrata | 33564 |
| 3 | Açores | Partido Popular | 5215 |
| 4 | Açores | Partido Comunista Português | 1612 |
| 5 | Açores | Bloco de Esquerda | 992 |
| 6 | Açores | Partido Democrático do Atlântico | 437 |
| 7 | Açores | Partido Comunista dos Trabalhadores Portugueses | 330 |
| 8 | Açores | Movimento Partido da Terra | 178 |
| 9 | Aveiro | Partido Socialista | 145575 |
| 10 | Aveiro | Partido Social Democrata | 138686 |
| 11 | Aveiro | Partido Popular | 49183 |
| 12 | Aveiro | Partido Comunista Português | 12797 |
| 13 | Aveiro | Bloco de Esquerda | 4676 |
| 14 | Aveiro | Partido Comunista dos Trabalhadores Portugueses | 1511 |
| 15 | Aveiro | Partido Popular Monárquico | 1148 |
| 16 | Aveiro | Partido Humanista | 968 |
| 17 | Aveiro | Movimento Partido da Terra | 847 |
| 18 | Aveiro | Partido Solidariedade Nacional | 660 |
| 19 | Beja | Partido Socialista | 39728 |
| 20 | Beja | Partido Comunista Português | 24077 |
| 21 | Beja | Partido Social Democrata | 12308 |
| 22 | Beja | Partido Popular | 3315 |

Figura 5: Resultado Query 2.b).

c) Qual o partido que, ao nível de freguesia, registou o maior número de votos? Indique a sigla do partido, o nome da freguesia e os votos correspondentes.

```

1 SELECT votacoes.partido, freguesias.nome, votacoes.votos
2 FROM votacoes, freguesias
3 WHERE freguesias.codigo = votacoes.freguesia
4 ORDER BY votacoes.votos DESC
5 FETCH FIRST 1 ROW ONLY;

```

Listing 9: Votação mais elevada numa freguesia

O resultado desta query é o seguinte:

| | PARTIDO | NOME | VOTOS |
|---|---------|---------------|-------|
| 1 | PS | Agualva-Cacem | 15188 |

Figura 6: Resultado Query 2.c).

d) Para cada distrito indique qual o seu nome e a designação e número de votos do partido que nele teve melhor votação.

```

1 SELECT nome, designacao, votos
2 FROM (
3     SELECT
4         distritos.nome,
5         partidos.designacao,
6         SUM(votacoes.votos) AS votos,
7         MAX(SUM(votacoes.votos)) OVER (PARTITION BY distritos.nome) AS
            max_votos
8     FROM distritos, concelhos, freguesias, votacoes, partidos
9     WHERE distritos.codigo = concelhos.distrito
10    AND concelhos.codigo = freguesias.concelho
11    AND freguesias.codigo = votacoes.freguesia
12    AND votacoes.partido = partidos.sigla
13    GROUP BY distritos.nome, partidos.designacao
14 )
15 WHERE votos = max_votos
16 ORDER BY votos DESC;

```

Listing 10: Partido mais votado por distrito

O resultado desta query é o seguinte:

| | NOME | DESIGNACAO | VOTOS |
|----|------------------|--------------------------|--------|
| 1 | Lisboa | Partido Socialista | 480410 |
| 2 | Porto | Partido Socialista | 440162 |
| 3 | Braga | Partido Socialista | 195602 |
| 4 | Setúbal | Partido Socialista | 170193 |
| 5 | Aveiro | Partido Socialista | 145575 |
| 6 | Santarém | Partido Socialista | 110326 |
| 7 | Coimbra | Partido Socialista | 109956 |
| 8 | Leiria | Partido Social Democrata | 99091 |
| 9 | Viseu | Partido Social Democrata | 90116 |
| 10 | Faro | Partido Socialista | 87162 |
| 11 | Castelo Branco | Partido Socialista | 63398 |
| 12 | Vila Real | Partido Social Democrata | 56507 |
| 13 | Madeira | Partido Social Democrata | 56302 |
| 14 | Viana do Castelo | Partido Socialista | 55132 |
| 15 | Açores | Partido Socialista | 49947 |
| 16 | Guarda | Partido Socialista | 44254 |
| 17 | Évora | Partido Socialista | 42257 |
| 18 | Beja | Partido Socialista | 39728 |
| 19 | Bragança | Partido Social Democrata | 36841 |
| 20 | Portalegre | Partido Socialista | 36545 |

Figura 7: Resultado Query 2.d).

Questão 3 - Negação

Análise de igual forma a questão “Quais os partidos que não concorreram no distrito de Lisboa.

```

1 SELECT designacao
2 FROM partidos
3 WHERE sigla NOT IN (
4     SELECT partidos.sigla
5     FROM partidos, listas, distritos
6     WHERE distritos.nome = 'Lisboa'
7         AND distritos.codigo = listas.distrito
8         AND listas.partido = partidos.sigla
9 );

```

Listing 11: Partidos que não concorreram em Lisboa

O resultado desta query é o seguinte:

| DESIGNACAO |
|------------------------------------|
| 1 Partido Democrático do Atlântico |

Figura 8: Resultado Query 3.

Questão 4

A pergunta “Houve algum partido a vencer em todos as freguesias de um concelho do distrito do Porto? Indique código do concelho e sigla do partido.” é de natureza universal. Compare do ponto de vista temporal e de plano de execução as estratégias da dupla negação e da contagem em três situações diferentes (só no contexto Z):

a) Sem vista para calcular o vencedor em cada freguesia.

```

1 SELECT c.codigo, p.sigla
2 FROM zconcelhos c
3 JOIN zdistritos d ON d.codigo = c.distrito
4 JOIN zpartidos p ON 1 = 1
5 WHERE d.nome = 'Porto'
6     AND NOT EXISTS (
7         SELECT 1
8         FROM zfreguesias f
9         WHERE f.concelho = c.codigo
10            AND NOT EXISTS (
11                SELECT 1
12                FROM zvotacoes v
13                WHERE v.freguesia = f.codigo
14                    AND v.partido = p.sigla
15                    AND v.votos = (
16                        SELECT MAX(v2.votos)
17                        FROM zvotacoes v2
18                        WHERE v2.freguesia = f.codigo
19                    )
20            )
21 );

```

Listing 12: Dupla negação.

```

1 SELECT c.codigo, v.partido
2 FROM zdistritos d
3 JOIN zconcelhos c ON d.codigo = c.distrito
4 JOIN zfreguesias f ON c.codigo = f.concelho
5 JOIN zvotacoes v ON f.codigo = v.freguesia
6 WHERE d.nome = 'Porto'
7     AND v.votos = (
8         SELECT MAX(v2.votos)
9         FROM zvotacoes v2
10        WHERE v2.freguesia = f.codigo
11    )
12 GROUP BY c.codigo, v.partido
13 HAVING COUNT(DISTINCT f.codigo) = (
14     SELECT COUNT(*)

```

```

15 FROM zfreguesias f2
16 WHERE f2.concelho = c.codigo
17 )
18 ORDER BY c.codigo, v.partido;

```

Listing 13: Contagem.

b) Com vista.

```

1 CREATE OR REPLACE VIEW vw_vencedores_freguesia AS
2 SELECT v.freguesia, v.partido
3 FROM zvotacoes v
4 WHERE v.votos = (
5     SELECT MAX(v2.votos)
6     FROM zvotacoes v2
7     WHERE v2.freguesia = v.freguesia
8 );

```

Listing 14: Criação da vista.

```

1 SELECT c.codigo, p.sigla
2 FROM zconcelhos c
3 JOIN zdistritos d ON d.codigo = c.distrito
4 JOIN zpartidos p ON 1 = 1
5 WHERE d.nome = 'Porto'
6 AND NOT EXISTS (
7     SELECT 1
8     FROM zfreguesias f
9     WHERE f.concelho = c.codigo
10    AND NOT EXISTS (
11        SELECT 1
12        FROM vw_vencedores_freguesia vw
13        WHERE vw.freguesia = f.codigo
14              AND vw.partido = p.sigla
15    )
16 );

```

Listing 15: Dupla negação com vista.

```

1 SELECT c.codigo, vw.partido
2 FROM zdistritos d
3 JOIN zconcelhos c ON d.codigo = c.distrito
4 JOIN zfreguesias f ON c.codigo = f.concelho
5 JOIN vw_vencedores_freguesia vw ON f.codigo = vw.freguesia
6 WHERE d.nome = 'Porto'
7 GROUP BY c.codigo, vw.partido
8 HAVING COUNT(DISTINCT f.codigo) = (
9     SELECT COUNT(*)
10    FROM zfreguesias f2
11    WHERE f2.concelho = c.codigo
12 )
13 ORDER BY c.codigo, vw.partido;

```

Listing 16: Contagem com vista.

c) Com vista materializada (eventualmente com índices).

```
1 CREATE MATERIALIZED VIEW mv_vencedores_freguesia
2 BUILD IMMEDIATE
3 REFRESH ON DEMAND
4 AS
5 SELECT v.freguesia, v.partido
6 FROM zvotacoes v
7 WHERE v.votos = (
8     SELECT MAX(v2.votos)
9     FROM zvotacoes v2
10    WHERE v2.freguesia = v.freguesia
11 );
```

Listing 17: Criação da vista materializada.

```
1 SELECT c.codigo, p.sigla
2 FROM zconcelhos c
3 JOIN zdistritos d ON d.codigo = c.distrito
4 JOIN zpartidos p ON 1 = 1
5 WHERE d.nome = 'Porto'
6    AND NOT EXISTS (
7        SELECT 1
8        FROM zfreguesias f
9        WHERE f.concelho = c.codigo
10       AND NOT EXISTS (
11           SELECT 1
12           FROM mv_vencedores_freguesia mv
13          WHERE mv.freguesia = f.codigo
14             AND mv.partido = p.sigla
15       )
16 );
```

Listing 18: Dupla negação com vista materializada.

```
1 SELECT c.codigo, mv.partido
2 FROM zdistritos d
3 JOIN zconcelhos c ON d.codigo = c.distrito
4 JOIN zfreguesias f ON c.codigo = f.concelho
5 JOIN mv_vencedores_freguesia mv ON f.codigo = mv.freguesia
6 WHERE d.nome = 'Porto'
7 GROUP BY c.codigo, mv.partido
8 HAVING COUNT(DISTINCT f.codigo) = (
9     SELECT COUNT(*)
10    FROM zfreguesias f2
11   WHERE f2.concelho = c.codigo
12 )
13 ORDER BY c.codigo, mv.partido;
```

Listing 19: Contagem com vista materializada.

Apesar das diferentes estratégias descritas acima, o resultado das várias queries é, como esperado, sempre o mesmo:

| | CODIGO | PARTIDO |
|---|--------|---------|
| 1 | 1304 | PS |
| 2 | 1308 | PS |

Figura 9: Resultado Query 4.

Questão 5

Compare os planos de execução da pesquisa “Quantos votos tiveram o PS e o PSD nos distritos 11, 15 e 17”, considerando no contexto Z.

- a) Com índices árvore-B em zconcelhos.districto e zvotacoes.partido.

Para esta alínea, foram utilizados os índices já criados e explicados anteriormente.

- b) Com índices bitmap.

Para esta alínea, os índices em cima foram excluídos e novos foram criados:

```

1 DROP INDEX idx_zconcelhos_districto;
2 DROP INDEX idx_zvotacoes_partido;
3
4 CREATE BITMAP INDEX bm_zconcelhos_districto ON zconcelhos(districto);
5 CREATE BITMAP INDEX bm_zvotacoes_partido ON zvotacoes(partido);

```

Listing 20: Criação dos índices bitmap.

A query utilizada, que foi a mesma para ambas as alíneas, foi a seguinte:

```

1 select zvotacoes.partido, zconcelhos.districto, sum (zvotacoes.votos) as
   votos
2 from zvotacoes, zconcelhos, zfreguesias
3 where zvotacoes.partido in ('PS', 'PPDPSD') and zconcelhos.districto in
   (11, 15, 17) and zconcelhos.codigo = zfreguesias.concelho and
   zfreguesias.codigo = zvotacoes.freguesia
4 group by zvotacoes.partido, zconcelhos.districto
5 order by votos desc

```

Listing 21: Votos do PS e PSD nos distritos 11, 15 & 17.

O resultado desta query, que é mesmo para ambas as alíneas, é o seguinte:

| | PARTIDO | DISTRITO | VOTOS |
|---|---------|----------|--------|
| 1 | PS | 11 | 480410 |
| 2 | PPDPSD | 11 | 307961 |
| 3 | PS | 15 | 170193 |
| 4 | PPDPSD | 15 | 70340 |
| 5 | PPDPSD | 17 | 56507 |
| 6 | PS | 17 | 50691 |

Figura 10: Resultado Query 5.

4 Resultados

Questão 1 - Seleção e junção

a)

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|--|-------------|---------|-------------|------|
| SELECT STATEMENT | | | | 14 |
| HASH JOIN | | | | 10 |
| Access Predicates | | | | 14 |
| XFREGUESIAS.CONCELHO=XCONCELHOS.CODIGO | | | | 10 |
| TABLE ACCESS | XCONCELHOS | FULL | 1 | 3 |
| Filter Predicates | | | | |
| XCONCELHOS.NOME='Azambuja' | | | | |
| TABLE ACCESS | XFREGUESIAS | FULL | 4241 | 7 |

Figura 11: Plano de Execução nos Ambientes X & Y.

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|--|-------------------------|------------------------|-------------|------|
| SELECT STATEMENT | | | | 14 |
| HASH JOIN | | | | 4 |
| Access Predicates | | | | 14 |
| ZFREGUESIAS.CONCELHO=ZCONCELHOS.CODIGO | | | | 4 |
| NESTED LOOPS | | | | 14 |
| NESTED LOOPS | | | | 14 |
| STATISTICS COLLECTOR | | | | |
| TABLE ACCESS | ZCONCELHOS | BY INDEX ROWID BATCHED | 1 | 2 |
| INDEX | IDX_CONCELHOS_NOME | RANGE SCAN | 1 | 1 |
| Access Predicates | | | | |
| ZCONCELHOS.NOME='Azambuja' | | | | |
| INDEX | IDX_FREGUESIAS_CONCELHO | RANGE SCAN | 14 | 1 |
| Access Predicates | | | | |
| ZFREGUESIAS.CONCELHO=ZCONCELHOS.CODIGO | | | | |
| TABLE ACCESS | ZFREGUESIAS | BY INDEX ROWID | 14 | 2 |
| TABLE ACCESS | ZFREGUESIAS | FULL | 14 | 2 |

Figura 12: Plano de Execução no Ambiente Z.

Análise de Resultados

As duas consultas SQL têm o mesmo objetivo: listar as freguesias pertencentes ao concelho “Azambuja”. A primeira usa diretamente o código do concelho (1103), enquanto a segunda faz uma junção com a tabela concelhos para filtrar pelo nome. Ambas retornam os mesmos resultados, mas com planos de execução e custos distintos.

No ambiente X, sem índices nem restrições, é utilizado um Hash Join com full table scans em freguesias e concelhos, resultando num custo total de 10.

No ambiente Y, embora existam primary keys e foreign keys, a ausência de índices nos campos usados na filtragem faz com que o plano de execução seja idêntico ao do ambiente X, mantendo também o custo em 10. Isto acontece porque não existem índices explícitos sobre os atributos utilizados nos filtros da consulta, nomeadamente sobre concelhos.nome e freguesias.concelho.

Já no ambiente Z, o desempenho melhora consideravelmente graças aos índices `IDX_CONCELHOS_NOME` e `IDX_FREGUESIAS_CONCELHO`, que permitem localizar os dados de forma seletiva. O índice `IDX_CONCELHOS_NOME` permite localizar rapidamente o concelho com nome “Azambuja” e `IDX_FREGUESIAS_CONCELHO` que possibilita obter diretamente as freguesias associadas a esse concelho, sem necessidade de varrer toda a tabela. A junção é feita com Nested Loops e os dados são acedidos por ROWID, reduzindo o custo total para 4.

b)

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|------------------------------------|-------------|-----------|-------------|------|
| SELECT STATEMENT | | | 9 | 10 |
| SORT | | ORDER BY | 9 | 10 |
| HASH JOIN | | | 9 | 9 |
| Access Predicates | | | | |
| AND | | | | |
| XDISTRITOS.CODIGO=XLISTAS.DISTRITO | | | | |
| XPARTIDOS.SIGLA=XLISTAS.PARTIDO | | | | |
| MERGE JOIN | | CARTESIAN | 12 | 6 |
| TABLE ACCESS | XDISTRITOS | FULL | 1 | 3 |
| Filter Predicates | | | | |
| XDISTRITOS.NOME='Lisboa' | | | | |
| BUFFER | | | | |
| TABLE ACCESS | XPARTIDOS | SORT | 12 | 3 |
| TABLE ACCESS | XLISTAS | FULL | 12 | 3 |
| TABLE ACCESS | XLISTAS | FULL | 182 | 3 |

Figura 13: Plano de Execução no Ambiente X.

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|------------------------------------|-----------------------------|----------------|-------------|------|
| SELECT STATEMENT | | | 9 | 9 |
| SORT | | ORDER BY | 9 | 9 |
| HASH JOIN | | | 9 | 8 |
| Access Predicates | | | | |
| YPARTIDOS.SIGLA=YLISTAS.PARTIDO | | | | |
| HASH JOIN | | | 9 | 5 |
| Access Predicates | | | | |
| YDISTRITOS.CODIGO=YLISTAS.DISTRITO | | | | |
| NESTED LOOPS | | | 9 | 5 |
| NESTED LOOPS | | | 9 | 5 |
| STATISTICS COLLECTOR | | | | |
| TABLE ACCESS | YDISTRITOS | FULL | 1 | 3 |
| Filter Predicates | | | | |
| YDISTRITOS.NOME='Lisboa' | | | | |
| INDEX | YLISTAS_DISTRITO_PARTIDO_PK | RANGE SCAN | 9 | 1 |
| Access Predicates | | | | |
| YDISTRITOS.CODIGO=YLISTAS.DISTRITO | | | | |
| TABLE ACCESS | YLISTAS | BY INDEX ROWID | 9 | 2 |
| TABLE ACCESS | YLISTAS | FULL | 9 | 2 |
| TABLE ACCESS | YPARTIDOS | FULL | 12 | 3 |

Figura 14: Plano de Execução no Ambiente Y.

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|------------------------------------|-----------------------------|------------------------|-------------|------|
| SELECT STATEMENT | | | 9 | 8 |
| SORT | | ORDER BY | 9 | 8 |
| HASH JOIN | | | 9 | 7 |
| Access Predicates | | | | |
| ZPARTIDOS.SIGLA=ZLISTAS.PARTIDO | | | | |
| HASH JOIN | | | 9 | 4 |
| Access Predicates | | | | |
| ZDISTRITOS.CODIGO=ZLISTAS.DISTRITO | | | | |
| NESTED LOOPS | | | 9 | 4 |
| NESTED LOOPS | | | 9 | 4 |
| STATISTICS COLLECTOR | | | | |
| TABLE ACCESS | ZDISTRITOS | BY INDEX ROWID BATCHED | 1 | 2 |
| INDEX | IDX_DISTRITOS_NOME | RANGE SCAN | 1 | 1 |
| Access Predicates | | | | |
| ZDISTRITOS.NOME='Lisboa' | | | | |
| INDEX | ZLISTAS_DISTRITO_PARTIDO_PK | RANGE SCAN | 9 | 1 |
| Access Predicates | | | | |
| ZDISTRITOS.CODIGO=ZLISTAS.DISTRITO | | | | |
| TABLE ACCESS | ZLISTAS | BY INDEX ROWID | 9 | 2 |
| TABLE ACCESS | ZLISTAS | FULL | 9 | 2 |
| TABLE ACCESS | ZPARTIDOS | FULL | 12 | 3 |

Figura 15: Plano de Execução no Ambiente Z.

Análise de Resultados

A consulta envolve as tabelas distritos, listas e partidos, com duas junções, um filtro por distritos.nome = 'Lisboa' e ordenação pelo número de mandatos.

No ambiente X, são feitos full table scans em todas as tabelas e junções por Hash Join, e, para satisfazer a condição distritos.nome = 'Lisboa', é necessário percorrer integralmente a tabela distritos. A query inclui ainda uma junção cartesiana inicial. O plano termina com uma ordenação e apresenta um custo total de 10.

No ambiente Y, as primary keys e foreign keys permitem ao otimizador reconhecer as relações, melhorando ligeiramente o plano com Nested Loops e a junção com a tabela listas já se realiza de forma mais eficiente, recorrendo à chave primária composta YLISTAS_DISTRITO_PARTIDO_PK, por meio de um Range Scan. A tabela distritos continua a ser percorrida na totalidade, e o custo total reduz-se ligeiramente para 9.

No ambiente Z, os índices otimizados (IDX_DISTRITOS_NOME e ZLISTA_DISTRITO_PARTIDO_PK) permitem localizar os dados de forma direta e eficiente. As junções são feitas com Nested Loops, os acessos com ROWID e o custo desce para 8, refletindo uma execução bem mais eficiente.

c)

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|--|-------------|-----------|-------------|------|
| SELECT STATEMENT | | | 1 | 45 |
| SORT | | AGGREGATE | 1 | |
| HASH JOIN | | | 166 | 45 |
| Access Predicates | | | | |
| XFREGUESIAS.CODIGO=XVOTACOES.FREGUESIA | | | | |
| HASH JOIN | | | 212 | 13 |
| Access Predicates | | | | |
| AND | | | | |
| XDISTRITOS.CODIGO=XCONCELHOS.DISTRITO | | | | |
| XCONCELHOS.CODIGO=XFREGUESIAS.CONCELHO | | | | |
| TABLE ACCESS | XCONCELHOS | FULL | 308 | 3 |
| MERGE JOIN | | CARTESIAN | 4241 | 10 |
| TABLE ACCESS | XDISTRITOS | FULL | 1 | 3 |
| Filter Predicates | | | | |
| XDISTRITOS.NOME='Lisboa' | | | | |
| BUFFER | | SORT | 4241 | 7 |
| TABLE ACCESS | XFREGUESIAS | FULL | 4241 | 7 |
| TABLE ACCESS | XVOTACOES | FULL | 3323 | 32 |
| Filter Predicates | | | | |
| XVOTACOES.PARTIDO='BE' | | | | |

Figura 16: Plano de Execução no Ambiente X.

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|----------------------|--------------------------------|--|-------------|------|
| SELECT STATEMENT | | | 1 | 45 |
| SORT | | AGGREGATE | 1 | |
| HASH JOIN | | | 166 | 45 |
| Access Predicates | | YFREGUESIAS.CODIGO=YVOTACOES.FREGUESIA | | |
| NESTED LOOPS | | | 166 | 45 |
| STATISTICS COLLECTOR | | | | |
| HASH JOIN | | | 212 | 13 |
| Access Predicates | | AND | | |
| | | YDISTritos.CODIGO=YCONCELHOS.DISTRITO | | |
| | | YCONCELHOS.CODIGO=YFREGUESIAS.CONCELHO | | |
| TABLE ACCESS | YCONCELHOS | FULL | 308 | 3 |
| MERGE JOIN | | CARTESIAN | 4241 | 10 |
| TABLE ACCESS | YDISTritos | FULL | 1 | 3 |
| Filter Predicates | | YDISTritos.NOME='Lisboa' | | |
| TABLE ACCESS | YFREGUESIAS | FULL | 4241 | 7 |
| TABLE ACCESS | YVOTACOES_PARTIDO_FREGUESIA_PL | UNIQUE SCAN | 4241 | 7 |
| INDEX | | | | |
| Access Predicates | | AND | | |
| | | YVOTACOES.PARTIDO='BE' | | |
| | | YFREGUESIAS.CODIGO=YVOTACOES.FREGUESIA | | |
| TABLE ACCESS | YVOTACOES | BY INDEX ROWID | 1 | 32 |
| TABLE ACCESS | YVOTACOES | FULL | 3323 | 32 |
| Filter Predicates | | YVOTACOES.PARTIDO='BE' | | |

Figura 17: Plano de Execução no Ambiente Y.

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|----------------------|-----------------------------|--|-------------|------|
| SELECT STATEMENT | | | 1 | 23 |
| SORT | | AGGREGATE | 1 | |
| HASH JOIN | | | 166 | 23 |
| Access Predicates | | ZFREGUESIAS.CODIGO=ZVOTACOES.FREGUESIA | | |
| NESTED LOOPS | | | 166 | 23 |
| STATISTICS COLLECTOR | | | | |
| HASH JOIN | | | 212 | 10 |
| Access Predicates | | ZCONCELHOS.CODIGO=ZFREGUESIAS.CONCELHO | | |
| NESTED LOOPS | | | 15 | 3 |
| NESTED LOOPS | | | 15 | 3 |
| TABLE ACCESS | ZDISTritos | BY INDEX ROWID BATCHED | 1 | 2 |
| INDEX | IDX_DISTritos_NOME | RANGE SCAN | 1 | 1 |
| Access Predicates | | ZDISTritos.NOME='Lisboa' | | |
| INDEX | IDX_CONCELHOS_DISTrito | RANGE SCAN | 15 | 0 |
| Access Predicates | | ZDISTritos.CODIGO=ZCONCELHOS.DISTRITO | | |
| TABLE ACCESS | ZCONCELHOS | BY INDEX ROWID | 15 | 1 |
| TABLE ACCESS | ZFREGUESIAS | FULL | 4241 | 7 |
| INDEX | IDX_VOTACOES_PARTIDO_FRE... | RANGE SCAN | 1 | 13 |
| Access Predicates | | AND | | |
| | | ZVOTACOES.PARTIDO='BE' | | |
| | | ZFREGUESIAS.CODIGO=ZVOTACOES.FREGUESIA | | |
| INDEX | IDX_VOTACOES_PARTIDO_FRE... | RANGE SCAN | 3323 | 13 |
| Access Predicates | | ZVOTACOES.PARTIDO='BE' | | |

Figura 18: Plano de Execução no Ambiente Z.

Análise de Resultados

Esta query envolve quatro tabelas com encadeamento de junções: distritos, concelhos, freguesias e votacoes, aplicando um filtro de seleção sobre o nome do distrito e o partido, (“Lisboa”) e (“BE”), e soma os votos correspondentes.

No ambiente X, o plano usa Hash Joins e Full Table Scans em todas as tabelas, aplicando os

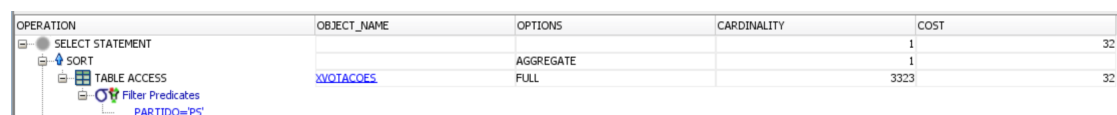
filtros apenas após as junções, o que gera elevado volume de dados processados e um custo total de 45.

No ambiente Y, apesar das primary keys e foreign keys, o plano mantém-se praticamente igual, com o mesmo custo. A falta de índices nos campos filtrados impede melhorias de desempenho.

No ambiente Z, O plano de execução utiliza Nested Loops combinados com Hash Joins, beneficiando dos índices `IDX_DISTRITOS_NOME` (para filtrar diretamente o distrito de Lisboa), `IDX_CONCELHOS_DISTRITO` (para encontrar os concelhos desse distrito), e `IDX_VOTACOES_PARTIDO_FREGUESIA` (para obter diretamente os registos de votações do partido “BE” nas freguesias relevantes). O custo é então reduzido para 23, tornando a execução significativamente mais eficiente.

Questão 2 - Agregação

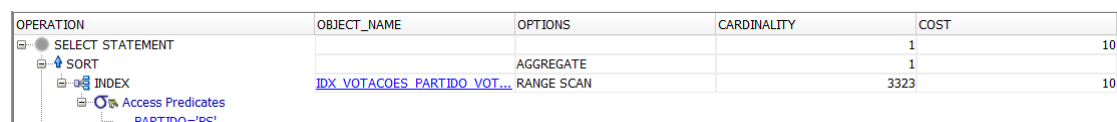
a)



| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|------------------|-------------|-----------|-------------|------|
| SELECT STATEMENT | | | | 32 |
| SORT | | AGGREGATE | 1 | 1 |
| TABLE ACCESS | VOTACOES | FULL | 3323 | 32 |

Filter Predicates: PARTIDO='PS'

Figura 19: Plano de Execução nos Ambientes X & Y.



| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|------------------|-----------------------------|------------|-------------|------|
| SELECT STATEMENT | | | | 10 |
| SORT | | AGGREGATE | 1 | 1 |
| INDEX | IDX_VOTACOES_PARTIDO_VOT... | RANGE SCAN | 3323 | 10 |

Access Predicates: PARTIDO='PS'

Figura 20: Plano de Execução no Ambiente Z.

Análise de Resultados

Esta é uma query simples, composta por uma filtragem sobre um único atributo (partido) na tabela votacoes e uma operação de agregação para somar os votos correspondentes.

No ambiente X, é feito um Full Table Scan da tabela, resultando num custo de 32, devido à leitura sequencial das 3323 linhas.

No ambiente Y, apesar da existência de primary keys e foreign keys, a ausência de índice sobre partido leva a um plano idêntico ao do ambiente X, mantendo o custo em 32.

Já no ambiente Z, o otimizador identifica a existência do índice `IDX_VOTACOES_PARTIDO_VOTOS`, que permite realizar um Range Scan diretamente sobre os registos que correspondem ao partido “PS”. Esta abordagem evita o scan completo da tabela, acedendo apenas aos dados necessários para a agregação. Como resultado, o custo total de execução é reduzido para 10.

b)

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|--|---------------|----------|-------------|------|
| SELECT STATEMENT | | | | 170 |
| SORT | | ORDER BY | | 170 |
| HASH | | GROUP BY | | 170 |
| HASH JOIN | | | | 170 |
| Access Predicates | | | | |
| XDISTritos.CODIGO=ITEM_2 | | | | |
| TABLE ACCESS | XDISTritos | FULL | 20 | 3 |
| HASH JOIN | | | | 170 |
| Access Predicates | | | | |
| ITEM_1=YPARTIDOS.SIGLA | | | | |
| TABLE ACCESS | YPARTIDOS | FULL | 12 | 3 |
| VIEW | SYS.VW_GBC_23 | | 170 | 44 |
| HASH JOIN | | GROUP BY | | 170 |
| Access Predicates | | | | |
| XFREGUESIAS.CODIGO=YVOTACOES.FREGUESIA | | | | |
| HASH JOIN | | | | 4241 |
| Access Predicates | | | | |
| XCONCELHOS.CODIGO=XFREGUESIAS.CONCELHO | | | | |
| TABLE ACCESS | XCONCELHOS | FULL | 308 | 3 |
| TABLE ACCESS | XFREGUESIAS | FULL | 4241 | 7 |
| TABLE ACCESS | XVOTACOES | FULL | 39874 | 32 |

Figura 21: Plano de Execução no Ambiente X.

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|--|----------------------|----------------|-------------|------|
| SELECT STATEMENT | | | | 170 |
| SORT | | ORDER BY | | 170 |
| HASH | | GROUP BY | | 170 |
| MERGE JOIN | | | | 170 |
| TABLE ACCESS | YPARTIDOS | BY INDEX ROWID | 12 | 2 |
| INDEX | YPARTIDOS.SIGLA_PK | FULL SCAN | 12 | 1 |
| SORT | | JOIN | | 170 |
| Access Predicates | | | | |
| ITEM_1=YPARTIDOS.SIGLA | | | | |
| Filter Predicates | | | | |
| ITEM_1=YPARTIDOS.SIGLA | | | | |
| VIEW | SYS.VW_GBC_23 | | 170 | 47 |
| HASH JOIN | | GROUP BY | | 170 |
| Access Predicates | | | | |
| YFREGUESIAS.CODIGO=YVOTACOES.FREGUESIA | | | | |
| HASH JOIN | | | | 4241 |
| Access Predicates | | | | |
| YCONCELHOS.CODIGO=YFREGUESIAS.CONCELHO | | | | |
| MERGE JOIN | | | | 308 |
| TABLE ACCESS | YDISTritos | BY INDEX ROWID | 20 | 2 |
| INDEX | YDISTritos.CODIGO_PK | FULL SCAN | 20 | 1 |
| SORT | | JOIN | | 308 |
| Access Predicates | | | | |
| YDISTritos.CODIGO=YCONCELHOS.DISTRITO | | | | |
| Filter Predicates | | | | |
| YDISTritos.CODIGO=YCONCELHOS.DISTRITO | | | | |
| TABLE ACCESS | YCONCELHOS | FULL | 308 | 3 |
| TABLE ACCESS | YFREGUESIAS | FULL | 4241 | 7 |
| TABLE ACCESS | YVOTACOES | FULL | 39874 | 32 |

Figura 22: Plano de Execução no Ambiente Y.

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|--|-------------------------|----------------|-------------|------|
| SELECT STATEMENT | | | | 170 |
| SORT | | ORDER BY | | 51 |
| HASH | | GROUP BY | | 170 |
| HASH JOIN | | | | 170 |
| Access Predicates | | | | 49 |
| ZDISTRITOS.CODIGO=ITEM_2 | | | | |
| VIEW | index\$_join\$_001 | | 20 | 2 |
| HASH JOIN | | | | |
| Access Predicates | | | | |
| ROWID=ROWID | | | | |
| INDEX | IDX_DISTritos_NOME | FAST FULL SCAN | 20 | 1 |
| INDEX | ZDISTRITOS_CODIGO_PK | FAST FULL SCAN | 20 | 1 |
| MERGE JOIN | | | 170 | 46 |
| TABLE ACCESS | ZPARTIDOS | BY INDEX ROWID | 12 | 2 |
| INDEX | ZPARTIDOS_SIGLA_PK | FULL SCAN | 12 | 1 |
| SORT | | JOIN | 170 | 44 |
| Access Predicates | | | | |
| ITEM_1=ZPARTIDOS.SIGLA | | | | |
| Filter Predicates | | | | |
| ITEM_1=ZPARTIDOS.SIGLA | | | | |
| VIEW | SYS.VW_GBC_23 | | 170 | 43 |
| HASH | | GROUP BY | 170 | 43 |
| HASH JOIN | | | 39874 | 42 |
| Access Predicates | | | | |
| ZFREGUESIAS.CODIGO=ZVOTACOES.FREGUESIA | | | | |
| HASH JOIN | | | 4241 | 9 |
| Access Predicates | | | | |
| ZCONCELHOS.CODIGO=ZFREGUESIAS.CONCELHO | | | | |
| NESTED LOOPS | | | 4241 | 9 |
| NESTED LOOPS | | | | |
| ST/ | | | | |
| index\$ _join\$ _002 | | | 308 | 2 |
| Access Predicates | | | | |
| ROWID=ROWID | | | | |
| INDEX | IDX_CONCELHOS_DISTrito | FAST FULL SCAN | 308 | 1 |
| INDEX | ZCONCELHOS_CODIGO_PK | FAST FULL SCAN | 308 | 1 |
| INDEX | IDX_FREGUESIAS_CONCELHO | RANGE SCAN | | |
| Access Predicates | | | | |
| ZCONCELHOS.CODIGO=ZFREGUESIAS.CONCELHO | | | | |
| TABLE | ZFREGUESIAS | BY INDEX ROWID | 14 | 7 |
| TABLE ACCESS | ZFREGUESIAS | FULL | 4241 | 7 |
| TABLE ACCESS | ZVOTACOES | FULL | 39874 | 32 |

Figura 23: Plano de Execução no Ambiente Z.

Análise de Resultados

Esta consulta junta cinco tabelas (distritos, concelhos, freguesias, votacoes e partidos), com agregação via GROUP BY e ordenação final.

No ambiente X, o plano de execução apresenta uma sucessão de Hash Joins e Full Table Scans, incluindo uma subconsulta entre votacoes, freguesias e concelhos, recorrendo a Full Scans, resultando num custo elevado de 53.

No ambiente Y, a presença de primary keys e foreign keys permite pequenas melhorias com Merge Joins e acessos por ROWID, mas como não há índices adicionais, as tabelas continuam a ser lidas por completo. O custo desce ligeiramente para 52.

No ambiente Z, o plano de execução não apresenta uma melhoria significativa. O otimizador consegue recorrer a Range Scans e Fast Full Index Scans para aceder diretamente aos dados relevantes, nomeadamente nos índices IDX_DISTritos_NOME, IDX_CONCELHOS_DISTrito, IDX_FREGUESIAS_CONCELHO e IDX_VOTACOES_PARTIDO_FREGUESIA. A subconsulta (vista) é tratada com operações de leitura por índices, e o uso de índices compostos permite juntar os dados de votacoes e freguesias de forma muito mais eficiente. Contudo o plano geral continua a ter o mesmo custo estimado porque o ganho do índice é mínimo e outras operações mais continuam a dominar o custo mantendo-o em 51.

c)

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|--|-------------|------------------|-------------|-------|
| SELECT STATEMENT | | | | 368 |
| VIEW | sys.null | | 1 | 368 |
| Filter Predicates | | | | |
| from\$_subquery\$_003.rowlimit_\$\$_rownumber <=1 | | | | |
| WINDOW | | SORT PUSHED RANK | | 39874 |
| Filter Predicates | | | | |
| ROW_NUMBER() OVER (ORDER BY INTERNAL_FUNCTION(XVOTACOES.VOTOS) DESC) <=1 | | | | |
| HASH JOIN | | | | 39 |
| Access Predicates | | | | |
| XFREGUESIAS.CODIGO=XVOTACOES.FREGUESIA | | | | |
| TABLE ACCESS | XFREGUESIAS | FULL | 4241 | 7 |
| TABLE ACCESS | XVOTACOES | FULL | 39874 | 32 |

Figura 24: Plano de Execução nos Ambientes X, Y & Z.

Análise de Resultados

Esta instrução envolve uma junção entre duas tabelas: votacoes e freguesias, com base no campo comum freguesia, seguida de uma ordenação decrescente pelo número de votos e uma limitação de resultados a uma única linha (FETCH FIRST 1 ROW ONLY). O objetivo é obter a linha correspondente ao registo com o maior número de votos.

Ao analisar o plano de execução desta query nos três ambientes (X, Y e Z), verifica-se que o custo total de execução é o mesmo em todos: 368, revelando que o otimizador escolheu uma abordagem semelhante em todos os contextos.

Em todos os ambientes, o plano baseia-se num Hash Join entre as tabelas votacoes e freguesias, sendo ambas acedidas por meio de varrimentos completos (Full Table Scans). Após a junção, é aplicada uma operação de windowing que utiliza a função analítica ROW_NUMBER() para ordenar os registos com base na coluna votos, em ordem decrescente. Em seguida, é filtrada apenas a primeira linha, correspondendo ao maior número de votos. Este tipo de processamento exige que o sistema percorra todos os dados antes de poder aplicar o ORDER BY e extrair a linha pretendida.

A razão para a igualdade dos custos entre os ambientes prende-se com a natureza da operação. Apesar de o ambiente Z possuir índices otimizados, estes não são utilizados neste caso concreto, pois a ordenação por votos em votacoes não pode ser aproveitada diretamente por índices sem cláusulas adicionais de filtragem (como WHERE partido = 'X', por exemplo), o número de votos não é chave primária nem faz parte de um índice unicamente ordenado por esse atributo e a consulta exige leitura completa da tabela votacoes para identificar o valor máximo, o que torna inevitável o Full Scan mesmo em ambiente com índices.

Assim, o custo elevado (368) reflete o esforço necessário para varrer as quase 40 mil linhas da tabela votacoes, juntá-las com freguesias, ordenar o resultado por votos e devolver apenas a linha com maior valor. Nenhum dos ambientes consegue otimizar significativamente esta operação sem alterar a estrutura da base de dados ou introduzir novas estratégias, como materialização de vistas com valores pré-computados ou índices específicos de ordenação.

d)

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|--|-------------|----------|-------------|-------|
| SELECT STATEMENT | | | | 170 |
| SORT | | ORDER BY | | 170 |
| VIEW | | | | 170 |
| Filter Predicates | | | | 53 |
| VOTOS=MAX_VOTOS | | | | |
| WINDOW | | | | 170 |
| HASH | | | | 170 |
| HASH JOIN | | | | 39874 |
| Access Predicates | | | | 49 |
| XVOTACOES.PARTIDO=XPARTIDOS.SIGLA | | | | |
| TABLE ACCESS | XPARTIDOS | FULL | 12 | 3 |
| HASH JOIN | | | | 39874 |
| Access Predicates | | | | 46 |
| XFREGUESIAS.CODIGO=XVOTACOES.FREGUESIA | | | | |
| HASH JOIN | | | | 4241 |
| Access Predicates | | | | 13 |
| XCONCELHOS.CODIGO=XFREGUESIAS.CONCELHO | | | | |
| HASH JOIN | | | | 308 |
| Access Predicates | | | | 6 |
| XDISTRITOS.CODIGO=XCONCELHOS.DISTRITO | | | | |
| TABLE ACCESS | XDISTRITOS | FULL | 20 | 3 |
| TABLE ACCESS | XCONCELHOS | FULL | 308 | 3 |
| TABLE ACCESS | XFREGUESIAS | FULL | 4241 | 7 |
| TABLE ACCESS | XVOTACOES | FULL | 39874 | 32 |

Figura 25: Plano de Execução no Ambiente X.

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|--|----------------------|----------------|-------------|-------|
| SELECT STATEMENT | | | | 170 |
| SORT | | ORDER BY | | 170 |
| VIEW | | | | 170 |
| Filter Predicates | | | | 53 |
| VOTOS=MAX_VOTOS | | | | |
| WINDOW | | | | 170 |
| HASH | | | | 170 |
| HASH JOIN | | | | 39874 |
| Access Predicates | | | | 49 |
| YVOTACOES.PARTIDO=YPARTIDOS.SIGLA | | | | |
| TABLE ACCESS | YPARTIDOS | FULL | 12 | 3 |
| HASH JOIN | | | | 39874 |
| Access Predicates | | | | 46 |
| YFREGUESIAS.CODIGO=YVOTACOES.FREGUESIA | | | | |
| HASH JOIN | | | | 4241 |
| Access Predicates | | | | 13 |
| YCONCELHOS.CODIGO=YFREGUESIAS.CONCELHO | | | | |
| MERGE JOIN | | | | 308 |
| TABLE ACCESS | YDISTRITOS | BY INDEX ROWID | 20 | 2 |
| INDEX | YDISTRITOS_CODIGO_PK | FULL SCAN | 20 | 1 |
| SORT | | JOIN | | 308 |
| Access Predicates | | | | 4 |
| YDISTRITOS.CODIGO=YCONCELHOS.DISTRITO | | | | |
| Filter Predicates | | | | |
| YDISTRITOS.CODIGO=YCONCELHOS.DISTRITO | | | | |
| TABLE ACCESS | YCONCELHOS | FULL | 308 | 3 |
| TABLE ACCESS | YFREGUESIAS | FULL | 4241 | 7 |
| TABLE ACCESS | YVOTACOES | FULL | 39874 | 32 |

Figura 26: Plano de Execução no Ambiente Y.

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|-------------------------|------------------------|--|-------------|------|
| SELECT STATEMENT | | | | 52 |
| SORT | | ORDER BY | 170 | 52 |
| VIEW | | | 170 | 51 |
| Filter Predicates | | VOTOS=MAX_VOTOS | | |
| WINDOW | | | | |
| HASH | | | | |
| HASH JOIN | | | 39874 | 47 |
| Access Predicates | | ZVOTACOES.PARTIDO=ZPARTIDOS.SIGLA | | |
| TABLE ACCESS | ZPARTIDOS | FULL | 12 | 3 |
| HASH JOIN | | | 39874 | 44 |
| Access Predicates | | ZFREGUESIAS.CODIGO=ZVOTACOES.FREGUESIA | | |
| NESTED LOOPS | | | 39874 | 44 |
| NESTED LOOPS | | | | |
| STATISTICS | | | | |
| HASH JOIN | | | 4241 | 11 |
| Access Predicates | | ZCONCELHOS.CODIGO=ZFREGUESIAS.CONCELHO | | |
| NESTED LOOPS | | | 4241 | 11 |
| STATISTICS | | | | |
| HASH JOIN | | | 308 | 4 |
| Access Predicates | | ZDISTRITOS.CODIGO=ZCONCELHOS.DISTRITO | | |
| index\$ join\$ 002 | | | 20 | 2 |
| Access Predicates | | ROWID=ROWID | | |
| IDX_DISTRITOS_NOME | | FAST FULL SCAN | 20 | 1 |
| ZDISTRITOS_CODIGO_PK | | FAST FULL SCAN | 20 | 1 |
| index\$ join\$ 003 | | | 308 | 2 |
| Access Predicates | | ROWID=ROWID | | |
| IDX_CONCELHOS_DISTRITO | | FAST FULL SCAN | 308 | 1 |
| ZCONCELHOS_CODIGO_PK | | FAST FULL SCAN | 308 | 1 |
| ZFREGUESIAS | | BY INDEX ROWID BATCHED | 14 | 7 |
| IDX_FREGUESIAS_CONCELHO | | RANGE SCAN | | |
| Access Predicates | | ZCONCELHOS.CODIGO=ZFREGUESIAS.CONCELHO | | |
| TABLE | ZFREGUESIAS | FULL | 4241 | 7 |
| INDEX | IDX_VOTACOES_FREGUESIA | RANGE SCAN | 39874 | 38 |
| Access Predicates | | ZFREGUESIAS.CODIGO=ZVOTACOES.FREGUESIA | | |
| TABLE ACCESS | ZVOTACOES | BY INDEX ROWID | 9 | 32 |
| TABLE ACCESS | ZVOTACOES | FULL | 39874 | 32 |

Figura 27: Plano de Execução no Ambiente Z.

Análise de Resultados

Esta consulta realiza múltiplas junções entre as tabelas distritos, concelhos, freguesias, votacoes e partidos, agregando os votos por distrito e partido. Em seguida, usa a função analítica MAX(...) OVER (PARTITION BY ...) para identificar os partidos mais votados em cada distrito, filtrando os resultados para mostrar apenas esses e ordenando por número de votos.

No ambiente X, sem restrições nem índices, o plano de execução recorre a Full Table Scans e Hash Joins em todas as tabelas. A agregação e a função analítica são aplicadas após as junções completas, o que gera um custo elevado de 54, refletindo o esforço de processar os cerca de 40 mil registros da tabela votacoes sem qualquer apoio estrutural.

No ambiente Y, apesar da presença de primary keys e foreign keys, a ausência de índices leva a um plano semelhante ao do ambiente X, com acesso apenas ligeiramente otimizado na tabela distritos. O custo mantém-se em 54, já que o volume de dados e a complexidade da consulta exigem leitura total.

Já no ambiente Z, a existência de índices nos campos usados nas junções e filtros permite otimizar os acessos - IDX_DISTRITOS_NOME, IDX_CONCELHOS_DISTRITO, IDX_FREGUESIAS_CONCELHO, IDX_VOTACOES_PARTIDO_FREGUESIA, entre outros. O plano utiliza Nested Loops, Hash

Joins e acessos por ROWID, reduzindo leituras desnecessárias. Apesar da complexidade da agregação e da função analítica, o custo desce para 52, mostrando que os índices melhoram a eficiência mesmo em consultas pesadas.

Questão 3

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|------------------------------------|--------------|-----------|-------------|------|
| SELECT STATEMENT | | | 3 | 12 |
| HASH JOIN | | ANTI | | 12 |
| Access Predicates | | | | |
| SIGLA=SIGLA | | | | |
| TABLE ACCESS | XPARTIDOS | FULL | 12 | 3 |
| VIEW | SYS.VW_NSQ_1 | | 9 | 9 |
| HASH JOIN | | SEMI | | 9 |
| Access Predicates | | | | |
| AND | | | | |
| XDISTRITOS.CODIGO=XLISTAS.DISTRITO | | | | |
| XLISTAS.PARTIDO=XPARTIDOS.SIGLA | | | | |
| MERGE JOIN | | CARTESIAN | 12 | 6 |
| TABLE ACCESS | XDISTRITOS | FULL | 1 | 3 |
| Filter Predicates | | | | |
| XDISTRITOS.NOME='Lisboa' | | | | |
| BUFFER | | SORT | 12 | 3 |
| TABLE ACCESS | XPARTIDOS | FULL | 12 | 3 |
| TABLE ACCESS | XLISTAS | FULL | 182 | 3 |

Figura 28: Plano de Execução no Ambiente X.

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|------------------------------------|-----------------------------|----------------|-------------|------|
| SELECT STATEMENT | | | 3 | 7 |
| MERGE JOIN | | ANTI | 3 | 7 |
| TABLE ACCESS | YPARTIDOS | BY INDEX ROWID | 12 | 2 |
| INDEX | YPARTIDOS_SIGLA_PK | FULL SCAN | 12 | 1 |
| SORT | | UNIQUE | 9 | 5 |
| Access Predicates | | | | |
| SIGLA=SIGLA | | | | |
| Filter Predicates | | | | |
| SIGLA=SIGLA | | | | |
| VIEW | SYS.VW_NSQ_1 | | 9 | 4 |
| HASH JOIN | | | 9 | 4 |
| Access Predicates | | | | |
| YDISTRITOS.CODIGO=YLISTAS.DISTRITO | | | | |
| NESTED LOOPS | | | 9 | 4 |
| STATISTICS COLLECTOR | | | | |
| TABLE ACCESS | YDISTRITOS | FULL | 1 | 3 |
| Filter Predicates | | | | |
| YDISTRITOS.NOME='Lisboa' | | | | |
| INDEX | YLISTAS_DISTRITO_PARTIDO_PK | RANGE SCAN | 9 | 1 |
| Access Predicates | | | | |
| YDISTRITOS.CODIGO=YLISTAS.DISTRITO | | | | |
| INDEX | YLISTAS_DISTRITO_PARTIDO_PK | FULL SCAN | 9 | 1 |

Figura 29: Plano de Execução no Ambiente Y.

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|------------------------------------|-----------------------------|------------------------|-------------|------|
| SELECT STATEMENT | | | | 6 |
| MERGE JOIN | | ANTI | 3 | 6 |
| TABLE ACCESS | ZPARTIDOS | BY INDEX ROWID | 12 | 2 |
| INDEX | ZPARTIDOS_SIGLA_PK | FULL SCAN | 12 | 1 |
| SORT | | UNIQUE | 9 | 4 |
| Access Predicates | | | | |
| SIGLA=SIGLA | | | | |
| Filter Predicates | | | | |
| SIGLA=SIGLA | | | | |
| VIEW | SYS.VW_NSO_1 | | 9 | 3 |
| HASH JOIN | | | 9 | 3 |
| Access Predicates | | | | |
| ZDISTRITOS.CODIGO=ZLISTAS.DISTRITO | | | | |
| NESTED LOOPS | | | 9 | 3 |
| STATISTICS COLLECTOR | | | | |
| TABLE ACCESS | ZDISTRITOS | BY INDEX ROWID BATCHED | 1 | 2 |
| INDEX | IDX_DISTRITOS_NOME | RANGE SCAN | 1 | 1 |
| Access Predicates | | | | |
| ZDISTRITOS.NOME='Lisboa' | | | | |
| INDEX | ZLISTAS_DISTRITO_PARTIDO_PK | RANGE SCAN | 9 | 1 |
| Access Predicates | | | | |
| ZDISTRITOS.CODIGO=ZLISTAS.DISTRITO | | | | |
| INDEX | ZLISTAS_DISTRITO_PARTIDO_PK | FULL SCAN | 9 | 1 |

Figura 30: Plano de Execução no Ambiente Z.

Análise de Resultados

A Query 3 identifica os partidos que não apresentaram listas no distrito de Lisboa, utilizando uma subconsulta com NOT IN que seleciona todas as siglas de partidos associadas a esse distrito. A consulta principal devolve as designações dos partidos cuja sigla não consta nesse subconjunto, recorrendo às tabelas partidos, listas e distritos.

No ambiente X, sem restrições nem índices, o plano de execução é pesado, com custo total de 12. Todas as tabelas são percorridas por Full Table Scans, e as junções são feitas por Hash Joins. A subconsulta é interpretada como uma view (SYS.VW_NSO_1) que junta listas, partidos e distritos, aplicando o filtro por “Lisboa” apenas depois da junção. A exclusão dos partidos é feita via Hash Anti Join, o que implica leitura completa da tabela partidos, tornando o plano pouco eficiente.

No ambiente Y, embora ainda sem índices adicionais, a existência de chaves primárias e estrangeiras permite ao otimizador usar estratégias mais eficientes. A tabela partidos é acedida via ROWID através do índice primário YPARTIDOS_SIGLA_PK, evitando varrimentos totais. A subconsulta é processada com Nested Loops e Range Scan sobre a chave composta de listas, reduzindo o custo para 7, graças à estrutura relacional formalizada.

No ambiente Z, com índices otimizados, o plano é ainda mais eficaz. O índice ZPARTIDOS_SIGLA_PK permite aceder à tabela partidos por ROWID, enquanto IDX_DISTRITOS_NOME identifica rapidamente o distrito de Lisboa, e ZLISTAS_DISTRITO_PARTIDO_PK localiza diretamente as listas relevantes. As junções usam Nested Loops e Hash Joins com acessos seletivos, o que reduz o custo total para 6.

Em resumo, a consulta mostra como a estrutura da base de dados afeta diretamente a eficiência: o ambiente X é o menos eficaz por falta de apoio estrutural; o ambiente Y melhora graças à existência de chaves; e o ambiente Z apresenta o melhor desempenho, graças aos índices criados estrategicamente, mesmo tratando-se de uma query com subconsulta e exclusão.

Questão 4

a)

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|--|-----------------------------|------------------------|-------------|------|
| SELECT STATEMENT | | | 185 | 282 |
| FILTER | | | | |
| Filter Predicates | | | | |
| NOT EXISTS (SELECT 0 FROM ZFREGUESIAS F WHERE NOT EXISTS (SELECT 0 FROM ZVOTACOES V WHERE V.VOTOS= (SELECT MAX(SYS_OP_UNDESCEND(SYS_OP_DESCEND(VOTOS))) FROM ZVOTACOES V2 WHERE V2.FREGUESIA=:B1 | | | | |
| HASH JOIN | | | 185 | 5 |
| Access Predicates | | | | |
| D.CODIGO=C.DISTRITO | | | | |
| MERGE JOIN | | CARTESIAN | 12 | 3 |
| TABLE ACCESS | ZDISTRITOS | BY INDEX ROWID BATCHED | 1 | 2 |
| INDEX | IDX_DISTRITOS_NOME | RANGE SCAN | 1 | 1 |
| Access Predicates | | | | |
| D.NOME='Porto' | | | | |
| BUFFER | | | | |
| INDEX | ZPARTIDOS_SIGLA_PK | FULL SCAN | 12 | 1 |
| VIEW | index\$_join\$.001 | | 12 | 1 |
| HASH JOIN | | | 308 | 2 |
| Access Predicates | | | | |
| ROWID=ROWID | | | | |
| INDEX | IDX_CONCELHOS_DISTRITO | FAST FULL SCAN | 308 | 1 |
| INDEX | ZCONCELHOS_CODIGO_PK | FAST FULL SCAN | 308 | 1 |
| FILTER | | | | |
| Filter Predicates | | | | |
| NOT EXISTS (SELECT 0 FROM ZVOTACOES V WHERE V.VOTOS= (SELECT MAX(SYS_OP_UNDESCEND(SYS_OP_DESCEND(VOTOS))) FROM ZVOTACOES V2 WHERE V2.FREGUESIA=:B1 | | | | |
| TABLE ACCESS | ZFREGUESIAS | BY INDEX ROWID BATCHED | 2 | 2 |
| INDEX | IDX_FREGUESIAS_CONCELHO | RANGE SCAN | 14 | 1 |
| Access Predicates | | | | |
| F.CONCELHO=:B1 | | | | |
| INDEX | IDX_VOTACOES_PARTIDO_FRE... | RANGE SCAN | 1 | 1 |
| Access Predicates | | | | |
| AND | | | | |
| V.PARTIDO=:B1 | | | | |
| V.FREGUESIA=:B2 | | | | |
| V.VOTOS= (SELECT MAX(SYS_OP_UNDESCEND(SYS_OP_DESCEND(VOTOS))) FROM ZVOTACOES V2 WHERE V2.FREGUESIA=:B3) | | | | |
| SORT | | AGGREGATE | 1 | |
| INDEX | IDX_VOTACOES_FREGUESIA_V... | RANGE SCAN | 9 | 2 |
| Access Predicates | | | | |
| V2.FREGUESIA=:B1 | | | | |

Figura 31: Plano de Execução da dupla negação.

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|-----------------------------------|---|------------------------|-------------|------|
| SELECT STATEMENT | | | 1 | 77 |
| SORT | | ORDER BY | 1 | 77 |
| FILTER | | | | |
| Filter Predicates | COUNT(DISTINCT \$vm_col_1)= (SELECT COUNT(*) FROM ZFREGUESIAS F2 WHERE F2.CONCELHO=:B1) | | | |
| SORT | | GROUP BY | 1 | 77 |
| VIEW | SYS.VM_NWVW_2 | | 11 | 76 |
| FILTER | | | | |
| Filter Predicates | V.VOTOS=MAX(SYS_OP_UNDESCEND(SYS_OP_DESCEND(VOTOS))) | | | |
| HASH | | GROUP BY | 11 | 76 |
| HASH JOIN | | | 18745 | 75 |
| Access Predicates | F.CODIGO=V.FREGUESIA | | | |
| HASH JOIN | | | 1994 | 42 |
| Access Predicates | V2.FREGUESIA=F.CODIGO | | | |
| HASH JOIN | | | 212 | 10 |
| Access Predicates | C.CODIGO=F.CONCELHO | | | |
| NESTED L | | | 15 | 3 |
| NESTE | | | 15 | 3 |
| TA4ZDISTRITOS | | BY INDEX ROWID BATCHED | 1 | 2 |
| IDX_DISTRITOS_NOME | | RANGE SCAN | 1 | 1 |
| Access Predicates | D.NOME='Porto' | | | |
| IND IDX_CONCELHOS_DISTRITO | | RANGE SCAN | 15 | 0 |
| Access Predicates | D.CODIGO=C.DISTRITO | | | |
| TABLE ZCONCELHOS | | BY INDEX ROWID | 15 | 1 |
| TABLE ACZ FREGUESIAS | | FULL | 4241 | 7 |
| INDEX IDX_VOTACOES_FREGUESIA_V... | | FAST FULL SCAN | 39874 | 32 |
| TABLE ACCESS ZVOTACOES | | FULL | 39874 | 32 |
| SORT | | AGGREGATE | 1 | |
| INDEX IDX_FREGUESIAS_CONCELHO | | RANGE SCAN | 14 | 1 |
| Access Predicates | F2.CONCELHO=:B1 | | | |

Figura 32: Plano de Execução da contagem.

b)

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|-------------------|--|------------------------|-------------|---------|
| SELECT STATEMENT | | | | 185 375 |
| FILTER | | | | |
| Filter Predicates | NOT EXISTS (SELECT 0 FROM ZFREGUESIAS F WHERE NOT EXISTS (SELECT 0 FROM UP202108753.ZVOTACOES V,UP202108753.ZVOTACOES V2 WHERE V2.FREGUESIA=:B | | | |
| HASH JOIN | | | 185 | 5 |
| Access Predicates | D.CODIGO=C.DISTRITO | | | |
| MERGE JOIN | | CARTESIAN | 12 | 3 |
| TABLE ACCESS | ZDISTRITOS | BY INDEX ROWID BATCHED | 1 | 2 |
| INDEX | IDX_DISTRITOS_NOME | RANGE SCAN | 1 | 1 |
| Access Predicates | D.NOME='Porto' | | | |
| BUFFER | | SORT | 12 | 1 |
| INDEX | ZPARTIDOS_SIGLA_PK | FULL SCAN | 12 | 1 |
| VIEW | index\$_join\$_001 | | 308 | 2 |
| HASH JOIN | | | | |
| Access Predicates | ROWID=ROWID | | | |
| INDEX | IDX_CONCELHOS_DISTRITO | FAST FULL SCAN | 308 | 1 |
| INDEX | ZCONCELHOS_CODIGO_PK | FAST FULL SCAN | 308 | 1 |
| FILTER | | | | |
| Filter Predicates | NOT EXISTS (SELECT 0 FROM UP202108753.ZVOTACOES V,UP202108753.ZVOTACOES V2 WHERE V2.FREGUESIA=:B1 AND V.FREGUESIA=:B2 AND V.PARTIDO=:B3 GROU | | | |
| TABLE ACCESS | ZFREGUESIAS | BY INDEX ROWID BATCHED | 2 | 2 |
| INDEX | IDX_FREGUESIAS_CONCELHO | RANGE SCAN | 14 | 1 |
| Access Predicates | F.CONCELHO=:B1 | | | |
| FILTER | | | | |
| Filter Predicates | V.VOTOS=MAX(SYS_OP_UNDESCEND(SYS_OP_DESCEND(VOTOS))) | | | |
| HASH | | GROUP BY | 1 | 2 |
| NESTED LOOPS | | | 7 | 2 |
| INDEX | IDX_VOTACOES_PARTIDO_FRE... | RANGE SCAN | 1 | 1 |
| Access Predicates | | | | |
| AND | V.PARTIDO=:B1 | | | |
| AND | V.FREGUESIA=:B2 | | | |
| INDEX | IDX_VOTACOES_FREGUESIA_V... | RANGE SCAN | 9 | 1 |
| Access Predicates | V2.FREGUESIA=:B1 | | | |

Figura 33: Plano de Execução da dupla negação com vista.

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|-------------------|--|------------------------|-------------|------|
| SELECT STATEMENT | | | 1 | 79 |
| SORT | | ORDER BY | 1 | 79 |
| FILTER | | | | |
| Filter Predicates | COUNT(DISTINCT F.CODIGO)=(SELECT COUNT(*) FROM ZFREGUESIAS F2 WHERE F2.CONCELHO=:B1) | | | |
| SORT | | GROUP BY | 1 | 79 |
| HASH JOIN | | SEMI | 1 | 77 |
| Access Predicates | | | | |
| AND | V.VOTOS=MAX(V2.VOTOS) SYS_OP_DESCEND(V.VOTOS)=SYS_OP_DESCEND(MAX(V2.VOTOS)) ITEM_1=V.FREGUESIA | | | |
| HASH JOIN | | | 1994 | 43 |
| Access Predicates | F.CODIGO=V.FREGUESIA | | | |
| NESTED LOOPS | | | 1994 | 43 |
| STATISTICS COL | | | | |
| HASH JOIN | | | 212 | 10 |
| Access Predicates | C.CODIGO=F.CONCELHO | | | |
| NESTED LOOPS | | | 15 | 3 |
| NESTED LOOPS | | | 15 | 3 |
| TABLE ACCESS | ZFREGUESIAS | BY INDEX ROWID BATCHED | 1 | 2 |
| INDEX | IND_FREGUESIAS_NOME | RANGE SCAN | 1 | 1 |
| Access Predicates | D.NOME='Porto' | | | |
| INDEX | IND_FREGUESIAS_DISTRITO | RANGE SCAN | 15 | 0 |
| Access Predicates | D.CODIGO=C.DISTRITO | | | |
| TABLE ACCESS | ZCONCELHOS | BY INDEX ROWID | 15 | 1 |
| TABLE ACCESS | ZFREGUESIAS | FULL | 4241 | 7 |
| INDEX | IND_VOTACOES_FREGUESIA | RANGE SCAN | 39874 | 38 |
| Access Predicates | F.CODIGO=V.FREGUESIA | | | |
| TABLE ACCESS | ZVOTACOES | BY INDEX ROWID | 9 | 32 |
| TABLE ACCESS | ZVOTACOES | FULL | 39874 | 32 |
| VIEW | SYS.VW_SO_1 | | 4241 | 34 |
| HASH | | GROUP BY | 4241 | 34 |
| INDEX | IND_VOTACOES_FREGUESIA_V... | FAST FULL SCAN | 39874 | 32 |
| SORT | | AGGREGATE | 1 | |
| INDEX | IND_FREGUESIAS_CONCELHO | RANGE SCAN | 14 | 1 |
| Access Predicates | F2.CONCELHO=:B1 | | | |

Figura 34: Plano de Execução da contagem com vista.

c)

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|--|-------------------------|------------------------|-------------|---------|
| SELECT STATEMENT | | | | 185 655 |
| FILTER | | | | |
| Filter Predicates | | | | |
| NOT EXISTS (SELECT 0 FROM ZFREGESIAS F WHERE NOT EXISTS (SELECT 0 FROM MV_VENCEDORES_FREGUESIA MV WHERE MV.FREGUESIA=:B1 AND MV.PARTIDO=:B2) | | | | |
| HASH JOIN | | | 185 | 5 |
| Access Predicates | | | | |
| D.CODIGO=C.DISTRITO | | | | |
| MERGE JOIN | | CARTESIAN | 12 | 3 |
| TABLE ACCESS | ZDISTRITOS | BY INDEX ROWID BATCHED | 1 | 2 |
| INDEX | IDX_DISTRITOS_NOME | RANGE SCAN | 1 | 1 |
| Access Predicates | | | | |
| D.NOME='Porto' | | | | |
| BUFFER | | | | |
| INDEX | ZPARTIDOS_SIGLA_PK | SORT | 12 | 1 |
| INDEX | index\$_join\$_001 | FULL SCAN | 12 | 1 |
| VIEW | | | 308 | 2 |
| HASH JOIN | | | | |
| Access Predicates | | | | |
| ROWID=ROWID | | | | |
| INDEX | IDX_CONCELHOS_DISTRITO | FAST FULL SCAN | 308 | 1 |
| INDEX | ZCONCELHOS_CODIGO_PK | FAST FULL SCAN | 308 | 1 |
| FILTER | | | | |
| Filter Predicates | | | | |
| NOT EXISTS (SELECT 0 FROM MV_VENCEDORES_FREGUESIA MV WHERE MV.FREGUESIA=:B1 AND MV.PARTIDO=:B2) | | | | |
| TABLE ACCESS | ZFREGESIAS | BY INDEX ROWID BATCHED | 2 | 2 |
| INDEX | IDX_FREGUESIAS_CONCELHO | RANGE SCAN | 14 | 1 |
| Access Predicates | | | | |
| F.CONCELHO=:B1 | | | | |
| MAT_VIEW ACCESS | MV_VENCEDORES_FREGUESIA | FULL | 1 | 5 |
| Filter Predicates | | | | |
| AND | | | | |
| MV.FREGUESIA=:B1 | | | | |
| MV.PARTIDO=:B2 | | | | |

Figura 35: Plano de Execução da dupla negação com vista materializada.

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|---|-------------------------|------------------------|-------------|------|
| SELECT STATEMENT | | | | 3 17 |
| SORT | | ORDER BY | 3 | 17 |
| FILTER | | | | |
| Filter Predicates | | | | |
| COUNT(DISTINCT F.CODIGO)=(SELECT COUNT(*) FROM ZFREGESIAS F2 WHERE F2.CONCELHO=:B1) | | | | |
| SORT | | | 3 | 17 |
| HASH JOIN | | GROUP BY | 218 | 15 |
| Access Predicates | | | | |
| F.CODIGO=MV.FREGUESIA | | | | |
| HASH JOIN | | | 212 | 10 |
| Access Predicates | | | | |
| C.CODIGO=F.CONCELHO | | | | |
| NESTED LOOPS | | | 15 | 3 |
| NESTED LOOPS | | | 15 | 3 |
| TABLE ACCESS | ZDISTRITOS | BY INDEX ROWID BATCHED | 1 | 2 |
| INDEX | IDX_DISTRITOS_NOME | RANGE SCAN | 1 | 1 |
| Access Predicates | | | | |
| D.NOME='Porto' | | | | |
| INDEX | IDX_CONCELHOS_DISTRITO | RANGE SCAN | 15 | 0 |
| Access Predicates | | | | |
| D.CODIGO=C.DISTRITO | | | | |
| TABLE ACCESS | ZCONCELHOS | BY INDEX ROWID | 15 | 1 |
| TABLE ACCESS | ZFREGESIAS | FULL | 4241 | 7 |
| MAT_VIEW ACCESS | MV_VENCEDORES_FREGUESIA | FULL | 4358 | 5 |
| SORT | | | | |
| INDEX | IDX_FREGUESIAS_CONCELHO | AGGREGATE | 1 | |
| Access Predicates | | | | |
| F2.CONCELHO=:B1 | | | 14 | 1 |

Figura 36: Plano de Execução da contagem com vista materializada.

Análise de Resultados

A questão 4 pergunta se houve algum partido que venceu em todas as freguesias de um concelho do distrito do Porto, representando uma consulta de natureza universal. Foram testadas duas estratégias: dupla negação e contagem, cada uma em três contextos: sem vista, com vista e com vista materializada.

No cenário **a) (sem vista)**, a **contagem** apresenta um custo de execução de **77**, enquanto a **dupla negação** tem um custo significativamente mais elevado de **282**. O custo elevado da dupla negação resulta do uso de subconsultas correlacionadas com `NOT EXISTS`, que implicam múltiplas junções e avaliações repetidas. Ainda que o sistema utilize índices como `IDX_DISTRITOS_NOME`, `IDX_CONCELHOS_DISTRITO`, `IDX_FREGUESIAS_CONCELHO` e `IDX_VOTACOES_PARTIDO_FREGUESIA_VOTOS`, a estrutura da consulta não permite aproveitar totalmente esses recursos.

No cenário **b) (com vista)**, contrariamente ao esperado, a **dupla negação com vista** apresenta um **custo ainda maior**, de **375**, enquanto a **contagem com vista** também sobe para **79**. Esta inversão sugere que a vista utilizada pode não ter sido aproveitada de forma eficiente pela estratégia de dupla negação — possivelmente por causa da forma como o otimizador reescreve a query. Mesmo com os dados pré-calculados, a verificação da condição universal implica acessos repetidos à vista, resultando em um plano de execução mais custoso.

No cenário **c) (com vista materializada)**, a **contagem com vista materializada** apresenta o melhor desempenho de todos os casos, com um custo de apenas **17**, beneficiando de acessos diretos aos dados já agregados e persistidos fisicamente. No entanto, a **dupla negação com vista materializada** é a pior de todas as versões testadas, com um custo extremamente elevado de **655**. Esta diferença acentuada demonstra que a materialização por si só não garante melhor desempenho, especialmente se a consulta continuar a depender de estruturas de subconsulta complexas e o otimizador não conseguir reescrever adequadamente os acessos.

A abordagem de **contagem** mostra-se consistentemente melhor, especialmente quando se recorre à materialização de vistas. Já a abordagem de **dupla negação** piora nos cenários com vista, o que reforça a importância de avaliar cuidadosamente a estratégia de formulação das queries, mesmo quando se utilizam estruturas auxiliares como vistas e índices.

Questão 5

a)

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|--|--|------------------------|-------------|------|
| SELECT STATEMENT | | | 5 | 44 |
| ↓ SORT | | ORDER BY | 5 | 44 |
| HASH | | GROUP BY | 5 | 44 |
| HASH JOIN | | | 997 | 42 |
| Access Predicates | ZFREGUESIAS.CODIGO=ZVOTACOES.FREGUESIA | | | |
| NESTED LOOPS | | | 997 | 42 |
| NESTED LOOPS | | | | |
| STATISTICS COLLECTOR | | | | |
| HASH JOIN | | | 636 | 9 |
| Access Predicates | ZCONCELHOS.CODIGO=ZFREGUESIAS.CONCELHO | | | |
| INLIST ITERATOR | | | | |
| TABLE ACCESS | ZCONCELHOS | BY INDEX ROWID BATCHED | 46 | 2 |
| INDEX | IDX_ZCONCELHOS_DISTRI | RANGE SCAN | 46 | 1 |
| Access Predicates | | | | |
| OR | | | | |
| ZCONCELHOS.DISTRITO=11 | | | | |
| ZCONCELHOS.DISTRITO=15 | | | | |
| ZCONCELHOS.DISTRITO=17 | | | | |
| TABLE ACCESS | ZFREGUESIAS | FULL | 4241 | 7 |
| INLIST ITERATOR | | | | |
| INDEX | ZVOTACOES.PARTIDO_FREGUE... | UNIQUE SCAN | | |
| Access Predicates | | | | |
| AND | | | | |
| OR | | | | |
| ZVOTACOES.PARTIDO='PPDPSD' | | | | |
| ZVOTACOES.PARTIDO='PS' | | | | |
| ZFREGUESIAS.CODIGO=ZVOTACOES.FREGUESIA | | | | |
| TABLE ACCESS | ZVOTACOES | BY INDEX ROWID | 2 | 32 |
| TABLE ACCESS | ZVOTACOES | FULL | 6646 | 32 |
| Filter Predicates | | | | |
| OR | | | | |
| ZVOTACOES.PARTIDO='PPDPSD' | | | | |
| ZVOTACOES.PARTIDO='PS' | | | | |

Figura 37: Plano de Execução com índices b-tree.

b)

| OPERATION | OBJECT_NAME | OPTIONS | CARDINALITY | COST |
|--|--|----------------|-------------|------|
| SELECT STATEMENT | | | 5 | 45 |
| SORT | | ORDER BY | 5 | 45 |
| HASH | | GROUP BY | 5 | 45 |
| HASH JOIN | | | 997 | 43 |
| Access Predicates | ZFREGUESIAS.CODIGO=ZVOTACOES.FREGUESIA | | | |
| NESTED LOOPS | | | 997 | 43 |
| NESTED LOOPS | | | | |
| STATISTICS COLLECTOR | | | | |
| HASH JOIN | | | 636 | 10 |
| Access Predicates | ZCONCELHOS.CODIGO=ZFREGUESIAS.CONCELHO | | | |
| TABLE ACCESS | ZCONCELHOS | FULL | 46 | 3 |
| Filter Predicates | | | | |
| OR | | | | |
| ZCONCELHOS.DISTRITO=11 | | | | |
| ZCONCELHOS.DISTRITO=15 | | | | |
| ZCONCELHOS.DISTRITO=17 | | | | |
| TABLE ACCESS | ZFREGUESIAS | FULL | 4241 | 7 |
| INLIST ITERATOR | | | | |
| INDEX | ZVOTACOES.PARTIDO.FREGUESIA | UNIQUE SCAN | | |
| Access Predicates | | | | |
| AND | | | | |
| OR | | | | |
| ZVOTACOES.PARTIDO='PPDPSD' | | | | |
| ZVOTACOES.PARTIDO='PS' | | | | |
| ZFREGUESIAS.CODIGO=ZVOTACOES.FREGUESIA | | | | |
| TABLE ACCESS | ZVOTACOES | BY INDEX ROWID | 2 | 32 |
| TABLE ACCESS | ZVOTACOES | FULL | 6646 | 32 |
| Filter Predicates | | | | |
| OR | | | | |
| ZVOTACOES.PARTIDO='PPDPSD' | | | | |
| ZVOTACOES.PARTIDO='PS' | | | | |

Figura 38: Plano de Execução com índices bitmap.

Análise de Resultados

Esta questão questiona qual é o número total de votos dos partidos **PS** e **PPDPSD** nos distritos 11, 15 e 17, realizando junções entre as tabelas **zvotacoes**, **zfreguesias** e **zconcelhos**.

Foram comparados dois planos de execução distintos no contexto Z, variando apenas o tipo de índices utilizados:

- a) Com índices **árvore-B** em **zconcelhos.distrato** e **zvotacoes.partido**;
- b) Com índices **bitmap** nas mesmas colunas.

No caso **a)**, com índices do tipo **árvore-B**, o plano apresenta um **custo total de 44**. O otimizador varre totalmente as tabelas **zfreguesias** e **zvotacoes**, mas aproveita a **IDX_CONCELHOS_DISTRITO** com *range scan* para filtrar os distritos, e a **IDX_VOTACOES_PARTIDO_FREGUESIA** para aceder diretamente aos votos por partido e freguesia.

Já no caso **b)**, com índices **bitmap**, o plano de execução é quase idêntico em estrutura, apresentando um **custo de 45**. A diferença de 1 unidade no custo é praticamente irrelevante, indicando que, neste cenário específico, o tipo de índice escolhido tem impacto mínimo. No entanto, os índices bitmap são geralmente mais eficazes quando há muitos valores repetidos (alta cardinalidade baixa), como é o caso da coluna **partido**, que contém apenas algumas siglas distintas, na tabela **ZVOTACOES**.

A comparação entre os dois planos mostra que, para esta questão, **o ganho com índices bitmap não é significativo**, devido à simplicidade da filtragem e à presença de boas junções. Ambos os tipos de índice resultam em planos eficientes, mas os bitmaps podem ser melhores

em casos com filtragens booleanas ou combinações múltiplas em colunas com poucos valores distintos.

5 Conclusão

A análise realizada ao longo deste trabalho permitiu evidenciar, de forma clara e fundamentada, a influência que as diferentes estratégias de otimização têm sobre o desempenho de queries SQL. Através da criação de três ambientes distintos — um desprovido de qualquer suporte (Ambiente X), outro com restrições de integridade (Ambiente Y) e um terceiro com índices otimizados (Ambiente Z) — foi possível avaliar o impacto isolado e combinado destas configurações sobre diversos tipos de interrogacoes, incluindo operações de seleção, junção, agregação, negação e consultas universais.

Verificou-se que o Ambiente X, por não possuir qualquer estrutura auxiliar, obriga a leituras completas das tabelas e a planos de execução baseados em Hash Joins, resultando em custos elevados de execução e menor eficiência.

Já o Ambiente Y, embora beneficie de restrições de integridade que permitem ao otimizador compreender melhor as relações entre as tabelas, apresenta apenas ganhos marginais quando comparado com o ambiente anterior, uma vez que continua a não dispor de acesso seletivo por índices.

Foi no Ambiente Z que se observaram os maiores benefícios ao nível da performance. A introdução de índices cuidadosamente escolhidos sobre colunas críticas (como chaves estrangeiras e campos frequentemente utilizados em filtros e ordenações) resultou numa redução significativa dos custos de execução em grande parte das queries. Isto verificou-se especialmente em interrogações com elevados volumes de dados ou com operações de agregação complexas, onde o uso de índices permitiu evitar full scans e aplicar estratégias de acesso seletivo mais eficientes, como Range Scans, Nested Loops e leitura por ROWID.

Importa ainda destacar que, em queries particularmente pesadas ou com lógica analítica (por exemplo, funções OVER), os ganhos de desempenho, embora presentes, foram mais limitados, revelando que em certos cenários o custo computacional associado à lógica da consulta pode sobrepor-se aos benefícios proporcionados pela estrutura da base de dados.

Em suma, este estudo confirma que a estruturação adequada de uma base de dados — através da definição de restrições de integridade e, sobretudo, da criação de índices relevantes, tem um impacto direto na eficiência e escalabilidade das interrogações SQL. Como tal, a adoção criteriosa destas práticas constitui uma componente essencial na conceção e manutenção de sistemas de bases de dados relacionais otimizados, especialmente em contextos reais com grandes volumes de dados e consultas complexas.