PROJETO DE BASES DE DADOS

Parte 4

Grupo 40 – Turno BD225179L03

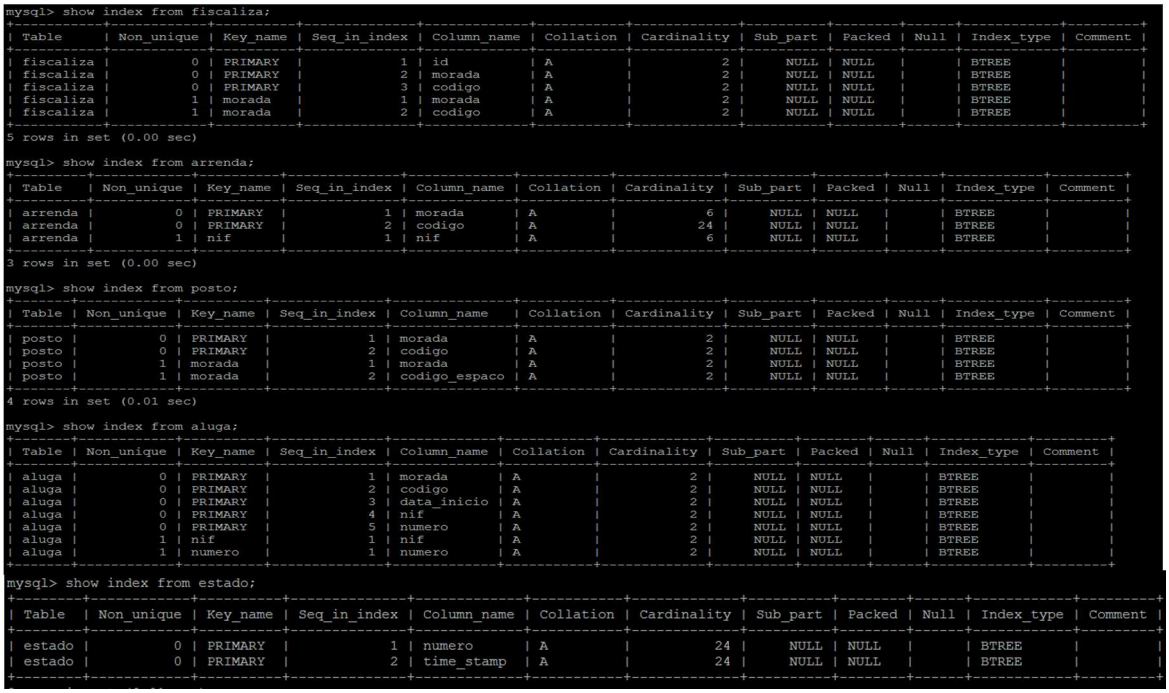
Rodrigo Lousada 81115 – 10 horas

Carlos Antunes 81525 – 10 horas

Nelson Duarte 67045 – 10 horas

- Índices

- Ao começar esta parte do projecto imediatamente fomos verificar quais os indices criados automaticamente pelo MySQL para as colunas utilizadas nas duas queries obtendo o seguinte resultado:



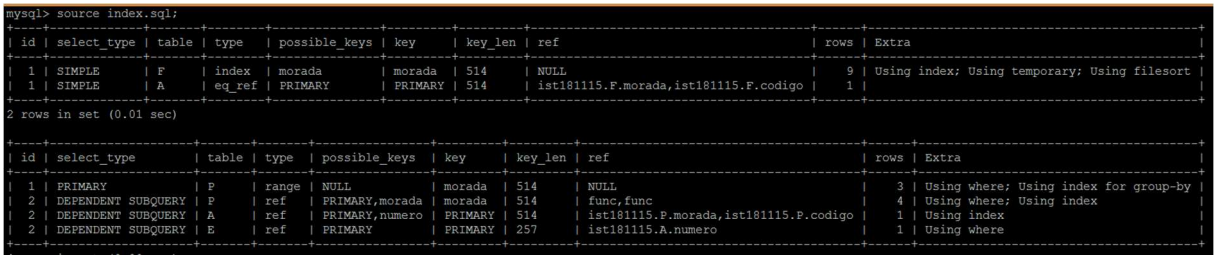
```
mysql> show index from fiscaliza;
+-----------+------------+----------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+
| Table     | Non_unique | Key_name | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment |
+-----------+------------+----------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+
| fiscaliza |          0 | PRIMARY  |            1 | id          | A         |           2 |     NULL | NULL   |      | BTREE      |         |
| fiscaliza |          0 | PRIMARY  |            2 | morada      | A         |           2 |     NULL | NULL   |      | BTREE      |         |
| fiscaliza |          0 | PRIMARY  |            3 | codigo      | A         |           2 |     NULL | NULL   |      | BTREE      |         |
| fiscaliza |          1 | morada   |            1 | morada      | A         |           2 |     NULL | NULL   |      | BTREE      |         |
| fiscaliza |          1 | morada   |            2 | codigo      | A         |           2 |     NULL | NULL   |      | BTREE      |         |
+-----------+------------+----------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+
5 rows in set (0.00 sec)

mysql> show index from arrenda;
+---------+------------+----------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+
| Table   | Non_unique | Key_name | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment |
+---------+------------+----------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+
| arrenda |          0 | PRIMARY  |            1 | morada      | A         |           6 |     NULL | NULL   |      | BTREE      |         |
| arrenda |          0 | PRIMARY  |            2 | codigo      | A         |          24 |     NULL | NULL   |      | BTREE      |         |
| arrenda |          1 | nif      |            1 | nif         | A         |           6 |     NULL | NULL   |      | BTREE      |         |
+---------+------------+----------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+
3 rows in set (0.00 sec)

mysql> show index from posto;
+-------+------------+----------+--------------+---------------+-----------+-------------+----------+--------+------+------------+---------+
| Table | Non_unique | Key_name | Seq_in_index | Column_name   | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment |
+-------+------------+----------+--------------+---------------+-----------+-------------+----------+--------+------+------------+---------+
| posto |          0 | PRIMARY  |            1 | morada        | A         |           2 |     NULL | NULL   |      | BTREE      |         |
| posto |          0 | PRIMARY  |            2 | codigo        | A         |           2 |     NULL | NULL   |      | BTREE      |         |
| posto |          1 | morada   |            1 | morada        | A         |           2 |     NULL | NULL   |      | BTREE      |         |
| posto |          1 | morada   |            2 | codigo_espaco | A         |           2 |     NULL | NULL   |      | BTREE      |         |
+-------+------------+----------+--------------+---------------+-----------+-------------+----------+--------+------+------------+---------+
4 rows in set (0.01 sec)

mysql> show index from aluga;
+-------+------------+----------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+
| Table | Non_unique | Key_name | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment |
+-------+------------+----------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+
| aluga |          0 | PRIMARY  |            1 | morada      | A         |           2 |     NULL | NULL   |      | BTREE      |         |
| aluga |          0 | PRIMARY  |            2 | codigo      | A         |           2 |     NULL | NULL   |      | BTREE      |         |
| aluga |          0 | PRIMARY  |            3 | data_inicio | A         |           2 |     NULL | NULL   |      | BTREE      |         |
| aluga |          0 | PRIMARY  |            4 | nif         | A         |           2 |     NULL | NULL   |      | BTREE      |         |
| aluga |          0 | PRIMARY  |            5 | numero      | A         |           2 |     NULL | NULL   |      | BTREE      |         |
| aluga |          1 | nif      |            1 | nif         | A         |           2 |     NULL | NULL   |      | BTREE      |         |
| aluga |          1 | numero   |            1 | numero      | A         |           2 |     NULL | NULL   |      | BTREE      |         |
+-------+------------+----------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+
mysql> show index from estado;
+--------+------------+----------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+
| Table  | Non_unique | Key_name | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment |
+--------+------------+----------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+
| estado |          0 | PRIMARY  |            1 | numero      | A         |          24 |     NULL | NULL   |      | BTREE      |         |
| estado |          0 | PRIMARY  |            2 | time_stamp  | A         |          24 |     NULL | NULL   |      | BTREE      |         |
+--------+------------+----------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+
```

- Depois para verificar quais os indices escolhidos automaticamnte pelo MySQL corremos o comando EXPLAIN (*query*) para ambas as queries, obtendo o seguinte output:



```
mysql> source index.sql;
+----+-------------+-------+-------+---------------+---------+---------+-----------------------------------------+------+-----------------------------------------------+
| id | select_type | table | type  | possible_keys | key     | key_len | ref                                     | rows | Extra                                         |
+----+-------------+-------+-------+---------------+---------+---------+-----------------------------------------+------+-----------------------------------------------+
|  1 | SIMPLE      | F     | index | morada        | morada  | 514     | NULL                                    |    9 | Using index; Using temporary; Using filesort |
|  1 | SIMPLE      | A     | ref   | PRIMARY       | PRIMARY | 514     | ist181115.F.morada,ist181115.F.codigo   |    1 |                                               |
+----+-------------+-------+-------+---------------+---------+---------+-----------------------------------------+------+-----------------------------------------------+
2 rows in set (0.01 sec)

+----+--------------------+-------+-------+-----------------+---------+---------+-----------------------------------------+------+--------------------------------------------+
| id | select_type        | table | type  | possible_keys   | key     | key_len | ref                                     | rows | Extra                                      |
+----+--------------------+-------+-------+-----------------+---------+---------+-----------------------------------------+------+--------------------------------------------+
|  1 | PRIMARY            | P     | range | NULL            | morada  | 514     | NULL                                    |    3 | Using where; Using index for group-by      |
|  2 | DEPENDENT SUBQUERY | P     | ref   | PRIMARY,morada  | morada  | 514     | func,func                               |    4 | Using where; Using index                   |
|  2 | DEPENDENT SUBQUERY | A     | ref   | PRIMARY,numero  | PRIMARY | 514     | ist181115.P.morada,ist181115.P.codigo   |    1 | Using index                                |
|  2 | DEPENDENT SUBQUERY | E     | ref   | PRIMARY         | PRIMARY | 257     | ist181115.A.numero                      |    1 | Using where                                |
+----+--------------------+-------+-------+-----------------+---------+---------+-----------------------------------------+------+--------------------------------------------+
```

A partir de conhecimentos teóricos tomámos em consideração as seguintes ideias iniciais:

- A estrutura baseada em bitmap apenas é benéfica caso seja aplicada a muitas colunas (ideia esta da qual não tinhamos a certeza até debatê-la com o professor numa aula de dúvidas)

- O MySQL escolhe sempre o indice mais eficiente. Não é possivel representar indices baseados em hash e bitmap.

- Para igualdades um índice hash-based melhorará sempre o desempenho nas alinhas do exercicio, sendo esta aplicado às colunas comparadas, o mesmo se aplica em Joins e NOT IN/IN.

1. Quais utilizadores cujos espaços foram fiscalizados sempre pelo mesmo fiscal?

```
select A.nif
from Arrenda A
      inner join Fiscaliza F
      on A.morada = F.morada
      and A.codigo = F.codigo
group by A.nif
having count(distinct F.id) = 1
```

a) Indique, justificando, que tipo de índice(s), sobre que atributo(s) e sobre que tabela(s) faria sentido criar de modo a acelerar a execução destas interrogações.

- Nesta solução o ficheiro ordenado por NIF aceleraria a execução. Ao criar um índice para esta situação temos um index-only plan sendo irrelevante se está agrupado.

- Para as igualdades de morada e código entre as tabelas arrenda e fiscaliza seria produtivo criar um index com estrutura baseada em hash e que os índices fossem compostos (morada, codigo).

b) Crie o(s) índice(s) em SQL, se necessário. Examine o plano de execução obtido para cada uma das queries e justifique.

- Criamos um index para o ID fiscaliza usando BTREE de forma a melhorar o desempenho do count dos ids sendo melhor uma vez que o MySQL não tinha criado nenhum automaticamente para esta situação.

- Criamos ainda os indices compostos para as tabelas arrenda e fiscaliza (atributos morada e codigo) no entanto usando BTREE o MySQL acaba por não os escolher pois não compensa, obtando assim pelo PRIMARY.

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	F	index	IndexFiscaliza	IndexFiscaliza	514	NULL	9	Using index; Using temporary; Using filesort
1	SIMPLE	A	eq_ref	PRIMARY,IndexArrenda	PRIMARY	514	ist181115.F.morada,ist181115.F.codigo	1	

2. Quais os espaços com postos que nunca foram alugados?

```
select distinct P.morada, P.codigo_espaco
from Posto P
where (P.morada, P.codigo_espaco) not in (
    select P.morada, P.codigo_espaco
    from Posto P
      natural join Aluga A
      natural join Estado E
    where E.estado = 'aceite')
```

a) Indique, justificando, que tipo de índice(s), sobre que atributo(s) e sobre que tabela(s) faria sentido criar de modo a acelerar a execução destas interrogações.

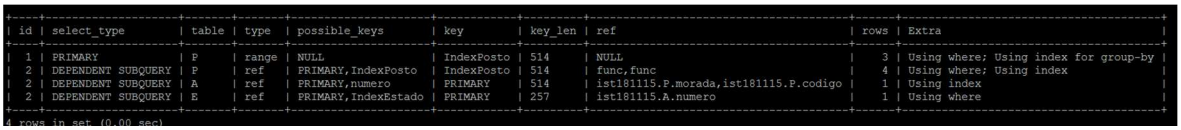
- Uma vez que a condição em estado é pouco seletiva e interrogação frequente então faz sentido criarmos um índice agrupado em estado.

- Também se justifica a estrutura hash composta para os atributos morada e codigo_espaco pois na comparação em not in confirmará imediamente no indice.

b) Crie o(s) índice(s) em SQL, se necessário. Examine o plano de execução obtido para cada uma das queries e justifique.

- Criamos o indice para estado no entanto mais uma vez não sendo Hash estes não vão ser escolhidos pois não apresentam um melhor resultado.

- Criamos ainda o indice composto para posto (atributos morada e codigo_espaco) usando BTREE sendo este escolhido e comprovando que embora não seja Hash apresenta uma melhoria.

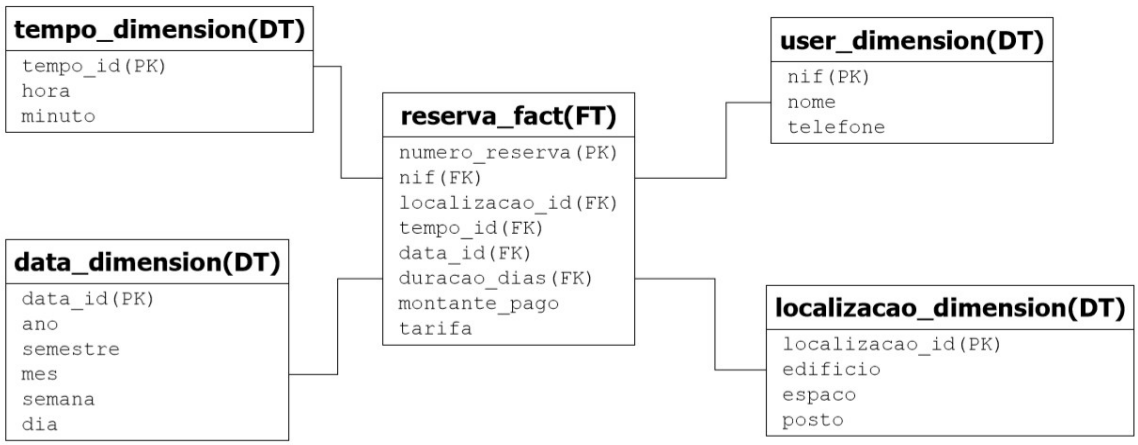


id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	PRIMARY	P	range	NULL	IndexPosto	514	NULL	3	Using where; Using index for group-by
2	DEPENDENT SUBQUERY	P	ref	PRIMARY,IndexPosto	IndexPosto	514	func,func	4	Using where; Using index
2	DEPENDENT SUBQUERY	A	ref	PRIMARY,numero	PRIMARY	514	ist181115.P.morada,ist181115.P.codigo	1	Using index
2	DEPENDENT SUBQUERY	E	ref	PRIMARY,IndexEstado	PRIMARY	257	ist181115.A.numero	1	Using where

4 rows in set (0.00 sec)

- Data Warehouse

1. - Para a criação da base de dados da data warehouse começámos primeiro por desenhar um esquema em estrela, para posteriormente implementar as tabelas (localizadas no ficheiro schema_warehouse.sql).



a) A tabela user_dimension possui exatamente os mesmos atributos que a tabela user da entrega anterior (schema disponibilizado pelo corpo docente da cadeira) sendo apenas necessário copiar de uma tabela para a outra.

```
INSERT INTO user_dimension  SELECT * FROM user;
```

b) A localizacao_dimension é a junção da tabela posto e espaço e a conversão das colunas morada, codigo e codigo_espaco em edificio, espaco e posto. Os espacos levam NULL na coluna posto, tendo ainda alterado este para uma string com um espaço para melhor leitura na alinea b deste exercicio.

```
INSERT INTO localizacao_dimension   SELECT concat(morada, '->', codigo) AS localizacao_id, morada AS edificio, codigo_espaco AS espaco, codigo AS posto
                                    FROM posto
                                    UNION
                                    SELECT concat(morada, '->', codigo) AS localizacao_id , morada AS edificio, codigo AS espaco, ' ' AS posto
                                    FROM espaco;
```

Os registos para as tabelas tempo_dimension e data_dimensions são gerados através de um procedure no qual fazemos dois ciclos:

c) O primeiro preenche todos os dias, semanas, meses e anos da que há entre 2016 e 2018 para a tabela data_dimension.
d) O segundo ciclo preenche todos os minutos e horas que há entre 2016 e 2018 para a tabela tempo dimension.

```
CREATE PROCEDURE load_data_time()
BEGIN
    DECLARE v_date DATE;
    DECLARE v_time DATETIME;
    DECLARE v_semestre INT;
    SET v_date = '2016-01-01';
    SET v_time = '2016-01-01 00:00:00';
    WHILE v_date < '2018-01-01' DO
        IF (MONTH(v_date) < 7)
            THEN SET v_semestre = 1;
            ELSE SET v_semestre = 2;
        END IF;

        INSERT INTO data_dimension(
            data_id,
            ano,
            semestre,
            mes,
            semana,
            dia
        ) VALUES (
            v_date,
            YEAR(v_date),
            v_semestre,
            MONTH(v_date),
            WEEK(v_date),
            DAY(v_date)
        );
        SET v_date = DATE_ADD(v_date, INTERVAL 1 DAY);
    END WHILE;

    WHILE v_time < '2016-01-02 00:00:00' DO
        INSERT INTO tempo_dimension(
            tempo_id,
            hora,
            minuto
        ) VALUES (
            DATE_FORMAT(v_time, '%H:%i'),
            HOUR(v_time),
            MINUTE(v_time)
        );
        SET v_time = DATE_ADD(v_time, INTERVAL 1 MINUTE);
    END WHILE;
END//
```

- A tabela reserva_fact contém como foreign keys as primary keys das tabelas user_dimension, localizacao_dimension, tempo_dimension e data_dimension. A inserção dos dados na tabela é feita com a seguinte query:

```
INSERT INTO reserva_fact            SELECT  numero,
                                        nif,
                                        concat(morada, '->', codigo) AS localizacao_id,
                                        DATE_FORMAT(data, '%H:%i') AS tempo_id,
                                        DATE(data) AS data_id,
                                        DATEDIFF(data_fim, data_inicio) AS duracao_dias,
                                        DATEDIFF(data_fim, data_inicio)*tarifa AS montante_pago,
                                        tarifa
                                FROM aluga NATURAL JOIN paga NATURAL JOIN oferta;
```

2. - Para a obter o cubo com o valor medio pago sobre a localização e data selecionamos o dia e mes da tabela date_dimension e o espaço e posto da localizacao_dimension para os quais calculamos a média do montante_pago. Depois agrupamos os valores por todos os subconjuntos possiveis do conjunto (dia, mes, espaço, posto).

```
SELECT dia, mes, espaco, posto, avg_montante_pago
FROM
    (SELECT dia, mes, espaco, posto, AVG(montante_pago) AS avg_montante_pago
    FROM reserva_fact NATURAL JOIN data_dimension NATURAL JOIN localizacao_dimension
    GROUP BY dia, mes, espaco, posto WITH ROLLUP

    UNION

    SELECT dia, mes, espaco, posto, AVG(montante_pago) AS avg_montante_pago
    FROM reserva_fact NATURAL JOIN data_dimension NATURAL JOIN localizacao_dimension
    GROUP BY mes, espaco, posto, dia WITH ROLLUP

    UNION

    SELECT dia, mes, espaco, posto, AVG(montante_pago) AS avg_montante_pago
    FROM reserva_fact NATURAL JOIN data_dimension NATURAL JOIN localizacao_dimension
    GROUP BY espaco, posto, dia, mes WITH ROLLUP

    UNION

    SELECT dia, mes, espaco, posto, AVG(montante_pago) AS avg_montante_pago
    FROM reserva_fact NATURAL JOIN data_dimension NATURAL JOIN localizacao_dimension
    GROUP BY posto, dia, mes, espaco WITH ROLLUP) AS A
ORDER BY dia, mes, espaco, posto;
```