Projecto de Bases de Dados, Parte 4

Rodolfo Cardoso (73861)
Pedro Torres (78742)
Rodrigo Bernardo (78942)

Grupo: 67

Turno: quinta-feira, 12h30

Professor: Gabriel Pestana

Data: 16/12/2016

Esforço: 6h para cada aluno

Índices

1. Quais os 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;
```

Caso a condição "A.morada = F.morada and A.codigo = F.codigo" seja pouco selectiva idealmente deveria-se ter um índice agrupado composto da tabela arrenda sobre os atributos . Caso esta condição seja muito selectiva deveria-se ter um indice agrupado composto sobre os atributos . Visto que os atributos morada e código são chaves primarias da tabela arrenda, esta já se encontra indexada em relação a estes dois atributos. Sendo assim não é vantajoso alterar estes índices no segundo caso, ou seja, quando se verifique que a condição "A.morada = F.morada and A.codigo = F.codigo" seja muito seletiva. Se for verificado o primeiro caso, ou seja, caso a condição "A.morada = F.morada and A.codigo = F.codigo" seja pouco seletiva, o desempenho da query pode ser melhorado com a criação do seguinte índice em MySQL:

```
CREATE UNIQUE INDEX arrenda_index ON arrenda (morada, codigo, nif);
```

Analisando o plano de execução desta query observa-se que a tabela arrenda utilizou os índices primários sendo estes do tipo B+tree.

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');
```

Visto que os joins "posto P NATURAL JOIN aluga A NATURAL JOIN estado E" reunem as tabelas apenas com a atributos chave primaria, estes já se encontram indexados. Para melhorar o desempenho pode ser criado um índice agrupado composto na tabela estado sobre os atributos caso a condição "E.estado = 'aceite' " seja pouco seletiva. Para criar este índice em MySQL usa-se:

```
CREATE INDEX estado_index ON estado (numero,estado);
```

Ao analisar o plano de execução desta query pode-se constatar que todas as tabelas utilizadas estão a ser acedidas com recurso a índices. Também se observa que na query principal são utilizados os índices da tabela gerada pela sub-query.

Data Warehouse

1. Cria na base de dados um esquema de uma estrela com informação sobre reservas.

```
drop table if exists f_reserva;
drop table if exists d_user;
drop table if exists d_local;
drop table if exists d_tempo;
drop table if exists d_data;

-- Data Warehouse

-------------------------------------------------------------------------------
-- Dimensoes

CREATE TABLE IF NOT EXISTS d_user(
    userid INT NOT NULL AUTO_INCREMENT,
    nif VARCHAR(9) NOT NULL,
    nome VARCHAR(80) NOT NULL,
    telefone VARCHAR(26) NOT NULL,
    PRIMARY KEY (userid));

CREATE TABLE IF NOT EXISTS d_local(
    localid INT NOT NULL AUTO_INCREMENT,
    codigo_posto VARCHAR(255) NOT NULL,
    codigo_espaco VARCHAR(255) NOT NULL,
    morada VARCHAR(255) NOT NULL,
    PRIMARY KEY (localid));

CREATE TABLE IF NOT EXISTS d_tempo(
    tempoid INT NOT NULL,
    minutos INT NOT NULL,
    horas INT NOT NULL,
    PRIMARY KEY (tempoid));

CREATE TABLE IF NOT EXISTS d_data(
    dataid INT NOT NULL,
    dia INT NOT NULL,
    semana INT NOT NULL,
    mes_numero INT NOT NULL,
    semestre INT NOT NULL,
    ano INT NOT NULL,
    PRIMARY KEY (dataid));

-------------------------------------------------------------------------------
-- Factos

CREATE TABLE IF NOT EXISTS f_reserva (
    userid INT NOT NULL,
    localid INT NOT NULL,
```

```
    tempoid INT NOT NULL,
    dataid INT NOT NULL,
    montante_pago INT NOT NULL,
    duracao_dias INT NOT NULL,
    PRIMARY KEY (userid,localid,tempoid,dataid),
    FOREIGN KEY (userid) REFERENCES d_user (userid) ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (localid) REFERENCES d_local (localid) ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (tempoid) REFERENCES d_tempo (tempoid) ON DELETE CASCADE ON UPDATE CASCADE,
    FOREIGN KEY (dataid) REFERENCES d_data (dataid) ON DELETE CASCADE ON UPDATE CASCADE);


-------------------------------------------------------------------------------
-- Populate das tabelas estaticas d_tempo e d_data

delimiter //

CREATE PROCEDURE load_tempo_dim()
BEGIN
    DECLARE v_full_time DATETIME;
    SET v_full_time = '2016-01-01 00:00:00';
    WHILE v_full_time < '2016-01-02 00:00:00' DO
    INSERT INTO d_tempo(
        tempoid,
        horas,
        minutos
        ) VALUES (
            HOUR(v_full_time) * 100 + MINUTE(v_full_time),
            HOUR(v_full_time),
            MINUTE(v_full_time)
        );
        SET v_full_time = DATE_ADD(v_full_time, INTERVAL 1 MINUTE);
    END WHILE;
END;
//

CREATE PROCEDURE load_data_dim()
BEGIN
    DECLARE v_full_date DATETIME;
    SET v_full_date = '2016-01-01 00:00:00';
    WHILE v_full_date < '2018-01-01 00:00:00' DO
    INSERT INTO d_data(
        dataid,
        ano,
        semestre,
        mes_numero,
        semana,
        dia
        ) VALUES (
            YEAR(v_full_date) * 10000 + MONTH(v_full_date)*100 + DAY(v_full_date),
            YEAR(v_full_date),
            IF(MONTH(v_full_date) < 7, 1, 2),
            MONTH(v_full_date),
```

```
            WEEK(v_full_date),
            DAY(v_full_date)
        );
        SET v_full_date = DATE_ADD(v_full_date, INTERVAL 1 DAY);
    END WHILE;
END;
//

delimiter ;

CALL load_tempo_dim;
CALL load_data_dim;
```

2. Escreva uma consulta OLAP para obter o cubo com valor médio pago sobre as dimensões localização e data.

```
SELECT
    SUM(FR.montante_pago) / COUNT(FR.montante_pago) AS Media,
    DL.localid                                      AS Localizacao,
    DT.dataid                                       AS Data
FROM
    f_reserva AS FR,
    d_local   AS DL,
    d_data    AS DT
WHERE
    FR.localid = DL.localid AND
    FR.dataid  = DT.dataid
GROUP BY
    DL.localid, DT.dataid WITH ROLLUP;
```