Projeto de Bases de Dados

Parte 4

Grupo 19 – Turno BD225179L09

Prof. Gabriel Pestana

81082 – Nuno Gonçalves (15 horas)

81205 – Alice Dourado (15 horas)

81500 – Rodrigo Rato (15 horas)

1. **Índices**

**Query #1 -** A)

Índices na tabela **arrenda**:

- Índice do tipo **HASH** para a coluna **morada**;

- Índice do tipo **HASH** para a coluna **codigo**;

- Índice do tipo **BTREE** para a coluna **nif**;

Índices na tabela **fiscaliza**:

- Índice do tipo **HASH** para a coluna **morada**;

- Índice do tipo **HASH** para a coluna **codigo**;

Os índices hash são colocados devido à comparação **a.morada = f.morada and a.codigo = f.código**, pois este tipo de índices são mais eficientes para operações de verificação de igualdade entre valores.

O índice BTREE sobre nif na tabela arrenda deve-se ao **group by a.nif**, que fica mais rápido ao percorrer a tabela por ordem, algo que é possível quando se tem este tipo de índice mas que é impossível de fazer num índice em hash onde determinar uma ordem de iteração da hash table é impossível.

**Query #1 -** B)

Para criar o índice BTREE do arrenda sobre a coluna nif usou-se a seguinte instrução:

**CREATE INDEX** nif\_idx **ON** arrenda(nif) **USING BTREE**;

Os índices HASH seriam criados com as seguintes instruções:

**CREATE INDEX** morada\_idx **ON** arrenda(morada) **USING HASH**;

**CREATE INDEX** codigo\_idx **ON** arrenda(codigo) **USING HASH;**

**CREATE INDEX** morada\_idx **ON** fiscaliza(morada) **USING HASH;**

**CREATE INDEX** codigo\_idx **ON** fiscaliza(codigo) **USING HASH;**

No entanto devido a limitações do *software* MySQL nao é possível criar índices hash uma vez que o storage engine usado nestas tabelas não suporta este tipo de índices. Também não é necessário criar o índice nif\_idx pois está associado a uma primary key da tabela logo é criado um índice automaticamente.

O plano de execução desta query foi o seguinte:





É possível observar que o índice btree que críamos não foi usado neste caso porque no resultado obtido há apenas um nif distinto com o conteúdo que está nas tabelas, mas possivelmente com um conteúdo diferente seria usado se fosse obtida uma maior quantidade de nifs a partir da query.

A primeira linha mostra que é usado um índice chamado morada que foi criado automaticamente na tabela fiscaliza, se fosse possível criar os índices hash que indicámos esses seriam usados em vez deste.

**Query #2 –** A)

Índice na tabela **estado**:

- Índice do tipo **HASH** para a coluna estado;

Índice na tabela **posto**:

- Índice do tipo **HASH** para as colunas morada, codigo\_espaco;

- Índice do tipo **HASH** para as colunas morada, codigo;

Índice na tabela **aluga**:

- Índice do tipo **HASH** para as colunas morada, codigo;

O índice do estado é criado porque, na query, é usada no **where** a condição **e.estado = ’aceite’** pelo que um índice do tipo hash será a melhor escolha para este caso pois este tipo de índice é mais eficiente para operações de verificação de igualdade entre valores.

O índice da tabela posto é devido ao facto de, na cláusula **where** da query efetuada, é usada a condição **(p.morada, p.codigo\_espaco) not in (…)**, o que implica que o valor irá ser comparado com todos os valores da tabela em segundo argumento e como isto é uma operação de verificação de igualdade o melhor índice para a situação é do tipo hash.

Os restantes índices são utilizados para melhorar o desempenho do **natural join** dentro da cláusula **not in,** que é efetuado através de comparações de valores de colunas com o mesmo nome, algo que é melhorado pelos hash indexes.

**Query #2 –** B)

Para criar se criarem os índices usaram-se a seguintes instruções:

**CREATE INDEX** estado\_idx **ON** estado(estado) **USING HASH**;

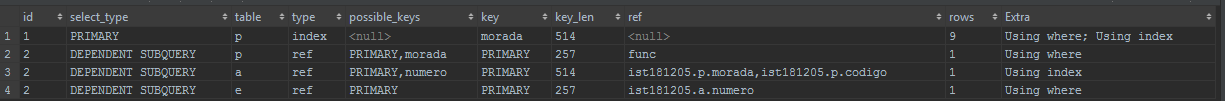
**CREATE INDEX** posto\_idx1 **ON** posto(morada, codigo\_espaco) **USING HASH**;

**CREATE INDEX** posto\_idx2 **ON** posto(morada, codigo) **USING HASH**;

**CREATE INDEX** aluga\_idx **ON** aluga(morada, codigo) **USING HASH**;

Não é possível criar estes índices em MySQL devido às limitações mencionadas anteriormente.

O plano de execução desta query foi o seguinte:



Para além dos índices das primary keys o outro índice usado é o índice ‘morada’ que foi criado automaticamente na tabela posto, mas se fosse possível criar/usar índices hash o nosso índice iria ser escolhido em detrimento do índice predefinido em btree.

1. **Data Warehouse**

**Schema para a base de dados OLAP:**

|  |  |
| --- | --- |
| **drop table if exists** reserva\_estrela; **drop table if exists** tempo\_dim; **drop table if exists** local\_dim; **drop table if exists** user\_dim; **drop table if exists** data\_dim;  **create table** tempo\_dim(  tempo **int not null unique**,  hora **int not null**,  minuto **int not null**,   **primary key**(tempo) );  **create table** data\_dim(  **data int not null unique**,  dia **int not null**,  semana **int not null**,  mes **int not null**,  semestre **int not null**,  ano **int not null**,  **primary key**(**data**) );  **create table** local\_dim(  morada\_codigo **varchar**(510) **not null**,  edificio **varchar**(255) **not null**,  espaco **varchar**(255) **not null**,  posto **varchar** (255),  **primary key**(morada\_codigo) ); | **create table** local\_dim(  morada\_codigo **varchar**(510) **not null**,  edificio **varchar**(255) **not null**,  espaco **varchar**(255) **not null**,  posto **varchar** (255),  **primary key**(morada\_codigo) );  **create table** user\_dim(  nif **varchar**(9) **not null unique**,  nome **varchar**(80) **not null**,  telefone **varchar**(26) **not null**,  **primary key**(nif) );  **create table** reserva\_estrela(  numero **varchar**(255) **not null unique**,  montante\_pago **numeric**(19,4) **not null**,  duracao\_dias **int not null**,   nif **varchar**(9) **not null**,  morada\_codigo **varchar**(510) **not null**,  tempo **int not null**,  **data int not null**,   **foreign key**(nif) **references** user\_dim(nif),  **foreign key**(morada\_codigo) **references** local\_dim(morada\_codigo),  **foreign key**(tempo) **references** tempo\_dim(tempo),  **foreign key**(**data**) **references** data\_dim(**data**),  **primary key**(numero, nif, morada\_codigo, tempo, **data**) ); |

**Query OLAP:**

|  |  |
| --- | --- |
| **SELECT** espaco, posto, dia, mes, *AVG*(montante\_pago) **AS** AVG\_MONTANTE\_PAGO **FROM** reserva\_estrela **NATURAL JOIN** data\_dim **NATURAL JOIN** local\_dim **GROUP BY** espaco, posto, **data UNION SELECT** espaco, posto, dia, **NULL**, *AVG*(montante\_pago) **AS** AVG\_MONTANTE\_PAGO **FROM** reserva\_estrela **NATURAL JOIN** data\_dim **NATURAL JOIN** local\_dim **GROUP BY** espaco,posto, dia **UNION SELECT** espaco, posto, **NULL**, mes, *AVG*(montante\_pago) **AS** AVG\_MONTANTE\_PAGO **FROM** reserva\_estrela **NATURAL JOIN** data\_dim **NATURAL JOIN** local\_dim **GROUP BY** espaco, posto, mes **UNION SELECT** espaco, **NULL**, dia, mes, *AVG*(montante\_pago) **AS** AVG\_MONTANTE\_PAGO **FROM** reserva\_estrela **NATURAL JOIN** data\_dim **NATURAL JOIN** local\_dim **GROUP BY** espaco, **data UNION SELECT NULL**, posto, dia, mes, *AVG*(montante\_pago) **AS** AVG\_MONTANTE\_PAGO **FROM** reserva\_estrela **NATURAL JOIN** data\_dim **NATURAL JOIN** local\_dim **WHERE** posto **IS NOT NULL GROUP BY** posto, **data UNION SELECT** espaco, posto, **NULL**, **NULL**, *AVG*(montante\_pago) **AS** AVG\_MONTANTE\_PAGO **FROM** reserva\_estrela **NATURAL JOIN** data\_dim **NATURAL JOIN** local\_dim **GROUP BY** espaco, posto **UNION SELECT NULL**, **NULL**, dia, mes, *AVG*(montante\_pago) **AS** AVG\_MONTANTE\_PAGO **FROM** reserva\_estrela **NATURAL JOIN** data\_dim **NATURAL JOIN** local\_dim **GROUP BY** (**data**) **UNION SELECT** espaco, **NULL**, dia, **NULL**, *AVG*(montante\_pago) **AS** AVG\_MONTANTE\_PAGO **FROM** reserva\_estrela **NATURAL JOIN** data\_dim **NATURAL JOIN** local\_dim **GROUP BY** espaco, dia **UNION** | **SELECT** espaco, **NULL**, **NULL**, mes, *AVG*(montante\_pago) **AS** AVG\_MONTANTE\_PAGO **FROM** reserva\_estrela **NATURAL JOIN** data\_dim **NATURAL JOIN** local\_dim **GROUP BY** espaco, mes **UNION SELECT NULL**, posto, **NULL**, mes, *AVG*(montante\_pago) **AS** AVG\_MONTANTE\_PAGO **FROM** reserva\_estrela **NATURAL JOIN** data\_dim **NATURAL JOIN** local\_dim **WHERE** posto **IS NOT NULL GROUP BY** posto, mes **UNION SELECT NULL**, posto, dia, **NULL**, *AVG*(montante\_pago) **AS** AVG\_MONTANTE\_PAGO **FROM** reserva\_estrela **NATURAL JOIN** data\_dim **NATURAL JOIN** local\_dim **WHERE** posto **IS NOT NULL GROUP BY** posto, dia **UNION SELECT** espaco, **NULL**, **NULL**, **NULL**, *AVG*(montante\_pago) **AS** AVG\_MONTANTE\_PAGO **FROM** reserva\_estrela **NATURAL JOIN** data\_dim **NATURAL JOIN** local\_dim **GROUP BY** espaco **UNION SELECT NULL**, posto, **NULL**, **NULL**, *AVG*(montante\_pago) **AS** AVG\_MONTANTE\_PAGO **FROM** reserva\_estrela **NATURAL JOIN** data\_dim **NATURAL JOIN** local\_dim **WHERE** posto **IS NOT NULL GROUP BY** posto **UNION SELECT NULL**, **NULL**, **NULL**, mes, *AVG*(montante\_pago) **AS** AVG\_MONTANTE\_PAGO **FROM** reserva\_estrela **NATURAL JOIN** data\_dim **NATURAL JOIN** local\_dim **GROUP BY** mes **UNION SELECT NULL**, **NULL**, dia, **NULL**, *AVG*(montante\_pago) **AS** AVG\_MONTANTE\_PAGO **FROM** reserva\_estrela **NATURAL JOIN** data\_dim **NATURAL JOIN** local\_dim **GROUP BY** dia **UNION SELECT NULL**, **NULL**, **NULL**, **NULL**, *AVG*(montante\_pago) **AS** AVG\_MONTANTE\_PAGO **FROM** reserva\_estrela **NATURAL JOIN** data\_dim **NATURAL JOIN** local\_dim; |

**Populate das tabelas do esquema em estrela:**

|  |  |
| --- | --- |
| **delimiter** // **drop procedure if exists** *buildtimedimension*// **create procedure** *buildtimedimension* () **begin  declare** begin\_time **TIME**;  **delete from** tempo\_dim;  **set** begin\_time = '00:00:00';  **while** begin\_time <= '23:59:00' **do  insert into** tempo\_dim(`tempo`,`hora`,`minuto`)   **values** (*HOUR*(begin\_time)\*10000 + *MINUTE*(begin\_time)\*100,*HOUR*(begin\_time),*MINUTE*(begin\_time));  **set** begin\_time = *ADDTIME*(begin\_time, '0:1:0.0');  **end while**; **end**// **delimiter** ;  **delimiter** // **drop procedure if exists** *builddatedimension*// **create procedure** *builddatedimension* () **begin  declare** begin\_date **DATE**;  **delete from** data\_dim;  **set** begin\_date = '2016-1-1';  **while** begin\_date <= '2017-12-31' **do  insert into** data\_dim(`data`,`dia`,`semana`,`mes`,`semestre`,`ano`)   **values** (*YEAR*(begin\_date) \* 10000 + *MONTH*(begin\_date)\*100 + *DAY*(begin\_date),*DAY*(begin\_date),*WEEK*(begin\_date),*MONTH*(begin\_date),*IF*(*MONTH*(begin\_date) < 7, 1, 2),*YEAR*(begin\_date));  **set** begin\_date = *DATE\_ADD*(begin\_date, **INTERVAL** 1 **DAY**);  **end while**; **end**// **delimiter** ; | **call** *buildtimedimension*();  **call** *builddatedimension*();  **INSERT INTO** user\_dim **SELECT** \* **from user**;  **INSERT INTO** local\_dim(`morada\_codigo`,`edificio`,`espaco`,`posto`) **SELECT** *CONCAT*(morada,codigo),morada,codigo\_espaco,codigo **FROM** posto;   **INSERT INTO** local\_dim(`morada\_codigo`,`edificio`,`espaco`) **SELECT** *CONCAT*(morada,codigo),morada,codigo **FROM** espaco;  **INSERT INTO** reserva\_estrela(`numero`,`nif`,`duracao\_dias`,`montante\_pago`,`morada\_codigo`,`data`,`tempo`) **SELECT** numero,nif,*datediff*(data\_fim,data\_inicio),*datediff*(data\_fim,data\_inicio)\*tarifa, *concat*(morada,codigo),*YEAR*(**data**) \* 10000 + *MONTH*(**data**)\*100 + *DAY*(**data**),  *HOUR*(**data**)\*10000 + *MINUTE*(**data**)\*100 **FROM** aluga **NATURAL JOIN** oferta **NATURAL JOIN** paga; |