Ricardo Bispo Amaral NUSP: 7693984
Pedro Toupitzen Specian NUSP: 8082640
Victor Caetano Da Silva NUSP: 9276999

Trabalho de LBD – 1º Semestre - 2017

Parte II – Artefato D

Requisito não-funcional – Necessidade de otimização

Especificação do requisito em texto

O processamento de consultas de motoristas e veículos exige muito tempo de execução devido à natureza das consultas. Portanto, é necessário otimizar as consultas para que elas possam recuperar e manipular informações em tempo satisfatório.

Solução textual em SQL padrão

O código abaixo cria uma visão com todos os dados de todos os motoristas e também dos veículos que cada um dirige, diferenciando motoristas de veículos leves (MOTORISTA_LEVE) e motoristas de veículos pesados (MOTORISTA_PESADO). Para cada motorista de veículo leve, são recuperados os dados dos furgões que eles dirigem, e para cada motorista de veículo pesado, são recuperados os dados dos caminhões que eles dirigem.

```
CREATE VIEW view_otimizacao AS

      SELECT * FROM FUNCIONARIO AS func

      INNER JOIN MOTORISTA AS mot ON mot.id_funcionario =
func.id_funcionario

      INNER JOIN MOTORISTA_LEVE AS motlev ON motlev.id_funcionario =
mot.id_funcionario

      INNER JOIN MOTORISTA_PESADO AS motpes ON motpes.id_funcionario =
mot.id_funcionario

      INNER JOIN dirige_furgao AS dir_fur ON dir_fur.id_funcionario =
motpes.id_funcionario

      INNER JOIN dirige_caminhao AS dir_cam ON dir_cam.id_funcionario
= motpes.id_funcionario

      INNER JOIN FURGAO AS furg ON furg.id_vei = dir_fur.id_vei

      INNER JOIN CAMINHAO AS cami ON cami.id_vei = dir_cam.id_vei

      INNER JOIN FROTA AS frotf ON frotf.id_vei = furg.id_vei

      INNER JOIN FROTA AS frotc ON frotc.id_vei = cami.id_vei;
```

Solução em código implementada dentro do PostgreSQL

```
CREATE VIEW view_otimizacao AS
```

```
      SELECT * FROM FUNCIONARIO AS func

      INNER JOIN MOTORISTA AS mot ON mot.id_funcionario =
func.id_funcionario

      INNER JOIN MOTORISTA_LEVE AS motlev ON motlev.id_funcionario =
mot.id_funcionario

      INNER JOIN MOTORISTA_PESADO AS motpes ON motpes.id_funcionario =
mot.id_funcionario

      INNER JOIN dirige_furgao AS dir_fur ON dir_fur.id_funcionario =
motpes.id_funcionario

      INNER JOIN dirige_caminhao AS dir_cam ON dir_cam.id_funcionario
= motpes.id_funcionario

      INNER JOIN FURGAO AS furg ON furg.id_vei = dir_fur.id_vei

      INNER JOIN CAMINHAO AS cami ON cami.id_vei = dir_cam.id_vei

      INNER JOIN FROTA AS frotf ON frotf.id_vei = furg.id_vei

      INNER JOIN FROTA AS frotc ON frotc.id_vei = cami.id_vei;
```

O código acima cria uma visão contendo todos os dados das tabelas FROTA, MOTORISTA_PESADO, CAMINHAO, MOTORISTA_LEVE, FURGAO, MOTORISTA e FUNCIONARIO. Ela, portanto, armazena os dados de todos os motoristas e os dados dos veículos associados a eles. Ao juntar tantas tabelas numa visão só, o custo de consultas que precisariam de múltiplas junções (por exemplo, recuperar os dados de um motorista e do veículo que ele dirige) diminui consideravelmente, pois certas junções deixariam de ser necessárias ao consultar a visão criada em vez das tabelas originais.

Consultas em SQL Padrão beneficiadas pela visão criada

Abaixo são listadas duas consultas que são beneficiadas pela visão que foi criada:

Consulta 1:

```
SELECT id_funcionario, nome, sobrenome, habilitacao_num,
habilitacao_categoria, id_vei, placa

FROM FUNCIONARIO AS f

INNER JOIN MOTORISTA AS m ON m.id_funcionario = f.id_funcionario

INNER JOIN dirige_caminhao AS dc ON dc.id_funcionario =
m.id_funcionario

INNER JOIN CAMINHAO AS c ON c.id_vei = dc.id_vei

INNER JOIN FROTA AS frt ON frt.id_vei = c.id_vei;
```

Consulta 2:

```
SELECT id_vei, placa, max_velo, num_portas, max_capacidade, max_itens

FROM FROTA
```

```
INNER JOIN FURGAO ON FURGAO.id_vei = FROTA.id_vei;
```

Discussão

Sugere-se que as ocorrências de contratações, promoções e demissões de funcionários, bem como alterações nos veículos que eles usam, não sejam tão comuns quanto as ocorrências de novos pedidos. Com isso, é seguro afirmar que as tabelas projetadas pela visão do artefato D possuem um fluxo de inserção, atualização e exclusão menor que o das tabelas projetadas pela visão do artefato C, mas possuem um fluxo de consultas maior que o da visão do artefato C. Tendo em vista que visões materializadas trazem vantagens para consultas, mas exigem atualização manual quando há alterações nas tabelas projetadas, seria vantajoso materializar a visão criada no artefato D.