Projeto BD - Parte 3

Grupo 89

L10

Francisco Regateiro

99175 - Alexandre Umbelino - 33,3% - 15 horas

92421 - André Rodrigues - 33,3% - 15 horas

87670 – João Martins - 33,3% - 15 horas

Restrições de Integridade

```
DROP TRIGGER IF EXISTS verifica_categoria ON tem_outra;

DROP TRIGGER IF EXISTS verifica_reposicao ON evento_reposicao;

DROP TRIGGER IF EXISTS verifica_produto_reposto ON planograma;


CREATE OR REPLACE FUNCTION verifica_categoria_trigger_proc()

RETURNS TRIGGER AS

$$

BEGIN

   IF (SELECT count(*) FROM tem_outra WHERE super_categoria = categoria) != 0
THEN

      RAISE EXCEPTION 'Categoria esta incluida em si propria';

   END IF;

   RETURN new;

  END;

$$ LANGUAGE plpgsql;


CREATE TRIGGER verifica_categoria AFTER INSERT OR UPDATE ON

tem_outra

EXECUTE PROCEDURE verifica_categoria_trigger_proc();


CREATE OR REPLACE FUNCTION verifica_reposicao_trigger_proc()

RETURNS TRIGGER AS

$$

BEGIN

   IF (SELECT tin FROM evento_reposicao, planograma

      WHERE planograma.ean = evento_reposicao.ean

      AND planograma.nro = evento_reposicao.nro

      AND planograma.num_serie = evento_reposicao.num_serie

      AND planograma.fabricante = evento_reposicao.fabricante

      AND planograma.unidades < evento_reposicao.unidades) != 0 THEN

         RAISE EXCEPTION 'Unidades repostas superiores ao que esta no
planograma';
```

```
    END IF;
    RETURN new;
   END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER verifica_reposicao AFTER INSERT OR UPDATE ON
evento_reposicao
FOR EACH ROW EXECUTE PROCEDURE verifica_reposicao_trigger_proc();

CREATE OR REPLACE FUNCTION verifica_produto_reposto_trigger_proc()
RETURNS TRIGGER AS
$$
BEGIN
    IF (SELECT count(planograma.ean) FROM planograma, tem_categoria, prateleira,
produto
        WHERE planograma.nro = prateleira.nro
        AND planograma.ean = produto.ean
        AND produto.cat != prateleira.nome AND produto.cat != prateleira.nome) != 0
THEN
            RAISE EXCEPTION 'Produto nao atribuido a prateleira certa';
    END IF;
    RETURN new;
END ;
$$ LANGUAGE plpgsql;
CREATE TRIGGER verifica_produto_reposto AFTER INSERT OR UPDATE ON
planograma
EXECUTE PROCEDURE verifica_produto_reposto_trigger_proc();
```

SQL

1) SELECT name FROM retalhista WHERE retalhista.tin =
(SELECT func.tin_max FROM (SELECT tin_max, count(n.tin_max)
AS tin_ FROM (SELECT DISTINCT tin AS tin_max, nome_cat,
count(responsavel_por.tin)
FROM responsavel_por GROUP BY tin, nome_cat ORDER BY tin
) AS n GROUP BY tin_max ORDER BY tin_max) AS func
WHERE func.tin_ = (SELECT max(tin_) from (SELECT tin_max,
count(n.tin_max)
AS tin_ FROM (SELECT DISTINCT tin AS tin_max, nome_cat,
count(responsavel_por.tin)
FROM responsavel_por GROUP BY tin, nome_cat ORDER BY tin
) AS n GROUP BY tin_max ORDER BY tin_max) AS func_2));

2) SELECT DISTINCT retalhista.name FROM responsavel_por, retalhista,
categoria_simples
WHERE retalhista.tin = responsavel_por.tin AND responsavel_por.nome_cat =
categoria_simples.nome;

3) SELECT ean FROM produto
WHERE ean NOT IN (SELECT ean FROM evento_reposicao);

4) SELECT ean FROM (SELECT ean, count(ean) AS count_ean FROM (SELECT
DISTINCT ean, tin FROM evento_reposicao
GROUP BY ean, tin ORDER BY ean)AS n GROUP BY ean) AS m
WHERE m.count_ean = 1;

Vistas

CREATE VIEW Vendas(ean, cat, ano, trimestre, mes, dia_mes, dia_semana, distrito, concelho, unidades) AS

SELECT tc.ean, tc.nome,

EXTRACT(YEAR FROM er.instante),

EXTRACT(QUARTER FROM er.instante),

EXTRACT(MONTH FROM er.instante),

EXTRACT(DAY FROM er.instante),

EXTRACT(DOW FROM er.instante),

pr.distrito, pr.concelho, er.unidades

FROM tem_categoria AS tc

INNER JOIN evento_reposicao AS er ON tc.ean = er.ean

INNER JOIN responsavel_por AS rp ON rp.num_serie = er.num_serie

INNER JOIN instalada_em AS ie ON ie.num_serie = rp.num_serie

```
    INNER JOIN ponto_de_retalho AS pr ON pr.nome = ie.local
WHERE tc.ean = er.ean;
```

Aplicação

Link para aplicação: https://web2.ist.utl.pt/ist187670/app.cgi/

Tendo em conta que a aplicação tem bastantes ficheiros e que não caberiam todos neste relatório, achamos importante fazer uma explicação sucinta do que foi feito.

Quando entramos, somos presenteados com um menu com 4 páginas, correspondendo ao que é pedido no enunciado:

- Categorias

Na página das Categorias, são listadas todas as categorias e subcategorias. Nessa mesma página é possível eliminar cada uma das categorias. Para além disso, se pretendermos adicionar uma outra categoria, é possível fazê-lo através do botão "Inserir Categoria". Para voltar para trás, existe sempre um botão de voltar para trás.

- Retalhistas

Na página dos Retalhistas, são listados todos os retalhistas. Mais uma vez é possível remover cada um dos retalhistas e também existe uma página para acrescentar Retalhistas, acessível através do botão "Inserir Retalhista".

- Eventos de reposição de IVM

Na página de Eventos de reposição de IVM, são apresentados os vários eventos pretendidos.

- Supercategorias

Na página de Supercategorias, são apresentados as várias supercategorias.

Análise de Dados

1)
```
SELECT CAST(ev.instante AS DATE) AS data, v.dia_semana AS
dia_da_semana, v.concelho, SUM(v.unidades) AS unidades_vendidas

FROM Vendas AS v, evento_reposicao AS ev

WHERE ev.instante BETWEEN '2021-01-01' AND '2021-12-31'

GROUP BY CUBE(ev.instante, v.dia_semana, v.concelho);
```

2)
```
SELECT v.distrito, v.concelho, v.cat AS categoria, v.dia_semana AS
dia_da_semana, SUM(v.unidades) AS unidades_vendidas

FROM Vendas AS v

GROUP BY CUBE(v.distrito, v.concelho, v.cat, v.dia_semana);
```