Projeto de Bases de Dados – Parte 4

Trabalho realizado por:

Nome	Número	Contribuição (%)	Esforço (horas)
Gonçalo Mateus	93713	33%	14
Guilherme Saraiva	93717	33%	14
Tomás Paiva	96961	33%	14

Grupo 37

Turno

Quarta 15:00 – 16:30

Docente

André Pereira

Restrições de Integridade (RI-análise, RI-100)

```
 create or replace function trigger_verifica_especialidade() returns trigger as $$
   begin

      if new.num_cedula is not null and new.num_doente is not null and new.data is not null then
         if ( select medico.especialidade
              from medico
              where medico.num_cedula = new.num_cedula ) = new.especialidade then
            return new;
         else
            raise exception 'A especialidade da analise nao corresponde a especialidade do medico.';
         end if;
      end if;

      return new;

   end;
$$ language plpgsql;

create trigger ana_esp_trg before update on analise for each row execute procedure
trigger_verifica_especialidade();

create trigger ana_esp_trg_i before insert on analise for each row execute procedure
trigger_verifica_especialidade();


create or replace function trigger_verifica_consultas_med() returns trigger as $$
   declare counter integer;
   begin

      select count(*) into counter
      from consulta c
      where c.num_cedula = new.num_cedula and c.nome_instituicao = new.nome_instituicao
        and extract(year from c.data) =  extract(year from new.data)
        and extract(week from c.data) = extract(week from new.data);

      if counter >= 100 then
         raise exception 'Um medico nao pode dar mais de 100 consultas por semana na mesma instituicao.';
      end if;

      return new;

   end;
$$ language plpgsql;

create trigger con_med_trg before update on consulta for each row execute procedure
trigger_verifica_consultas_med();

create trigger con_med_trg_i before insert on consulta for each row execute procedure
trigger_verifica_consultas_med();
```

Indexes

```
-- 1)
/*Esta query pode ser otimizada criando um índice hash
 sobre o num_cedula da consulta. Deste modo, a comparação
 direta "num_doente = <um_valor>" torna-se mais eficiente.
 A utilização de índices hash é preferida pois apenas se
 trata de uma simples igualdade. */

create index num_cedula_index on consulta using hash(num_cedula);


-- 2)
/*Para este caso, em semelhança à alínea acima, é necessário criar um
  índice para organizar a coluna "especialidade" com uma Hash Table
  pois torna-se mais eficiente a comparação "especialidade = “Ei”".
 */

create index especialidade_index on medico using hash(especialidade);


-- 3)
/* Nesta alínea, uma vez que apenas temos duas linhas por bloco e os médicos estão uniformemente distribuídos
por especialidade, isto é, o numero de médicos de cada especialidade é o mesmo, implica que se iriam fazer muitos
acessos aos blocos do disco.
   Para evitar esta situação, ordenam-se os médicos por especialidade usando uma btree de modo a que médicos
com a mesma especialidade fiquem juntos reduzindo assim os acessos aos blocos do disco. Para alem disto, a btree é
mais eficiente para tabelas de grande dimensão e portanto, é a melhor escolha para este caso.
 */

create index especialidade_index2 on medico using btree(especialidade);


-- 4)
/* Neste caso, para tornar a comparacao "consulta.num_celula = medico.num_celula" mais eficiente,
   criam-se duas Hash Tables, uma na tabela Consulta e outra na tabela Medico. Para a condição
   consulta.data BETWEEN ‘data_1’ AND ‘data_2’ deve-se organizar a coluna da data
   na tabela consulta e, por isso, utiliza-se uma btree pelo que é a mais eficiente
   para procura de ranges, neste caso, o BETWEEN.
 */

create index num_cedula_indexc on consulta using hash(num_cedula);
create index num_cedula_indexm on medico using hash(num_cedula);
create index data_index on consulta using btree(data);
```

Modelo Multidimensional

```
drop table if exists d_tempo cascade ; drop table if exists d_instituicao cascade ;
drop table if exists f_presc_venda cascade ; drop table if exists f_analise cascade ;

create table d_tempo (
   id_tempo serial not null,
   dia int not null,
   dia_da_semana int not null,
   semana int not null,
   mes int not null,
   trimestre int not null,
   ano int not null,
   primary key (id_tempo)
);
create table d_instituicao (
   id_inst serial not null,
   nome varchar(50) not null,
   tipo varchar(50) not null,
   num_regiao int not null,
   num_concelho int not null,
   foreign key (nome) references instituicao(nome) on delete cascade on update cascade,
   foreign key (num_regiao) references regiao(num_regiao) on delete cascade on update cascade,
   foreign key (num_concelho) references concelho(num_concelho) on delete cascade on update cascade,
   primary key (id_inst)
);
create table f_presc_venda (
   id_presc_venda serial not null,
   id_medico int not null,
   num_doente int not null,
   id_data_registo int not null,
   id_inst int not null,
   substancia varchar(50) not null ,
   quant int not null,
   foreign key (id_presc_venda) references prescricao_venda(num_venda) on delete cascade on update cascade,
   foreign key (id_medico) references medico(num_cedula) on delete cascade on update cascade ,
   foreign key (id_data_registo) references d_tempo(id_tempo) on delete cascade on update cascade ,
   foreign key (id_inst) references d_instituicao(id_inst) on delete cascade on update cascade ,
   primary key (id_presc_venda, id_data_registo, id_inst)
);
create table f_analise (
   id_analise serial not null,
   id_medico int not null ,
   num_doente int not null ,
   id_data_registo int not null,
   id_inst int not null,
   nome varchar(50) not null ,
   quant int not null ,
   foreign key (id_analise) references analise(num_analise) on delete cascade on update cascade ,
   foreign key (id_medico) references medico(num_cedula) on delete cascade on update cascade ,
   foreign key (id_data_registo) references d_tempo(id_tempo) on delete cascade on update cascade ,
   foreign key (id_inst) references d_instituicao(id_inst) on delete cascade on update cascade ,
   primary key (id_analise, id_data_registo, id_inst)
);
```

ETL de carregamento

```
insert into d_tempo(dia, dia_da_semana, semana, mes, trimestre, ano)
   select extract(day from data_registo) as dia,
        extract(dow from data_registo) as dia_da_semana,
        extract(week from data_registo) as semana,
        extract(month from data_registo) as mes,
        (extract(month from data_registo) + 1) / 3 as trimestre,
        extract(year from data_registo) as ano
     from analise
   union
   select extract(day from data) as dia,
        extract(dow from data) as dia_da_semana,
        extract(week from data) as semana,
        extract(month from data) as mes,
        (extract(month from data) + 1) / 3 as trimestre,
        extract(year from data) as ano
     from prescricao_venda
   order by dia, dia_da_semana, semana, mes, trimestre, ano;

insert into d_instituicao(nome, tipo, num_regiao, num_concelho)
   select nome, tipo, num_regiao, num_concelho from instituicao;

insert into f_presc_venda(id_presc_venda, id_medico, num_doente, id_data_registo, id_inst, substancia, quant)
   select num_venda as id_presc_venda, pv.num_cedula as id_medico, pv.num_doente, id_tempo as
id_data_registo,
        id_inst, pv.substancia, p.quant
   from prescricao_venda pv
      inner join medico m on pv.num_cedula = m.num_cedula
      inner join d_tempo dt on dt.dia = extract(day from pv.data)
                  and dt.mes = extract(month from pv.data)
                  and dt.ano = extract(year from pv.data)
      inner join prescricao p on pv.num_cedula = p.num_cedula
                    and pv.num_doente = p.num_doente
                    and pv.data = p.data
                    and pv.substancia = p.substancia
      inner join consulta c on p.num_cedula = c.num_cedula
                  and p.num_doente = c.num_doente  and p.data = c.data
      inner join instituicao i on c.nome_instituicao = i.nome
      inner join d_instituicao di on i.nome = di.nome
   group by num_venda, pv.num_cedula, pv.num_doente, id_tempo, id_inst, pv.substancia, p.quant;

insert into f_analise(id_analise, id_medico, num_doente, id_data_registo, id_inst, nome, quant)
   select num_analise as id_analise, a.num_cedula as id_medico, a.num_doente, id_tempo as id_data_registo,
id_inst, a.nome, quant
   from analise a
      inner join consulta c on c.num_cedula = a.num_cedula and c.num_doente = a.num_doente and c.data = a.data
      inner join d_tempo dt on dt.dia = extract(day from a.data_registo)
                  and dt.mes = extract(month from a.data_registo)
                  and dt.ano = extract(year from a.data_registo)
      inner join instituicao i on i.nome = c.nome_instituicao
      inner join d_instituicao di on i.nome = di.nome
   group by a.num_analise, a.num_cedula, a.num_doente, id_tempo, id_inst, a.nome, quant;
```

***Queries* OLAP**

1)

```
select especialidade, ano, mes, count(*) as ttl_anls_glic
from f_analise fa inner join d_tempo dt on dt.id_tempo = fa.id_data_registo
                inner join medico m on fa.id_medico = m.num_cedula
where fa.nome = 'Analise Glicemica' and ano between 2017 and 2020
group by cube (especialidade, ano, mes) order by (especialidade, ano, mes);
```

2)

```
select distinct t_total.num_concelho, t_total.mes, t_total.dia_da_semana, t_total.substancia, quant_total,
            cast(total_presc_day as float) / cast(4 as float) as media_dia_da_semana
from (
      select dt.mes, dt.dia_da_semana, fpv.substancia, count(*) as total_presc_day
      from f_presc_venda fpv inner join d_instituicao di on di.id_inst = fpv.id_inst
         inner join d_tempo dt on dt.id_tempo = fpv.id_data_registo
      where di.num_regiao = 3 and dt.trimestre = 1 and dt.ano = 2020
      group by rollup (dt.mes, dt.dia_da_semana), fpv.substancia order by (mes, dia_da_semana)
   ) as t_media
   inner join (
      select di.num_concelho, dt.mes, dt.dia_da_semana, fpv.substancia, sum(fpv.quant) as quant_total
      from f_presc_venda fpv inner join d_instituicao di on di.id_inst = fpv.id_inst
         inner join d_tempo dt on dt.id_tempo = fpv.id_data_registo
      where di.num_regiao = 3 and dt.trimestre = 1 and dt.ano = 2020
      group by rollup (di.num_concelho, dt.mes, dt.dia_da_semana), fpv.substancia order by (mes, dia_da_semana)
   ) as t_total on t_media.mes = t_total.mes and t_total.dia_da_semana = t_media.dia_da_semana and
t_total.substancia = t_media.substancia
group by rollup (t_total.num_concelho, t_total.mes, t_total.dia_da_semana), t_total.substancia, quant_total,
total_presc_day
```

Nota:

1. Junto com os files pedidos na E4, vai um populate novo para testar as queries OLAP.
2. Para a primeira query olap poderíamos ter usado o grouping sets em vez do cube para obter os totais parciais de cada coluna, mas preferimos apresentar a informação com mais detalhe e, portanto, usámos o group by cube.