DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

Relatório Projeto BDDAD

Scripts SQL para Base de Dados

Turma 2DB
1180017 - Gabriel Pelosi
1171250 – Rogério Alves
1151352 – João Mata

Docente
Angelo Manuel Rego E Silva Martins

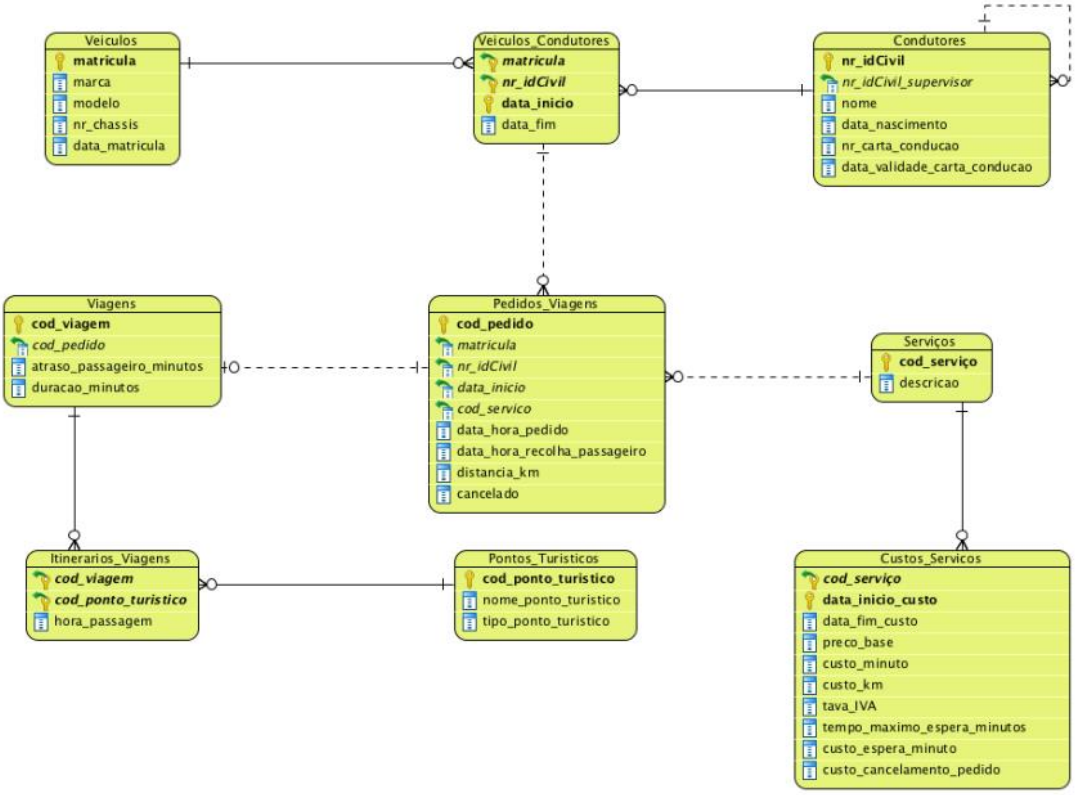
Unidade Curricular
Base de Dados (BDDAD)

Porto, 11 de Outubro 2019

isep Instituto Superior de Engenharia do Porto

Introdução

Este projeto foi desenvolvido por alunos do 2º ano do curso de Engenharia Informática do ISEP relativamente a diciplina Base de Dados,que aborda a linguagem de consulta estruturada SQL. O trabalho pratico teve como objetivo criar tabelas de acordo com o modelo relacional abaixo, inserir e consultar os dados das respectivas tabelas criadas através da plataforma Oracle SQL Developer,para assim, uma empresa de transporte indivudual poder registrar dados relativos as suas viagens.



Resultados e Justificações

A)

Apresente o nome dos condutores que menos vezes esteve associado a veículos da marca 'Toyota';

```
with Condutores_minimo_Toyotas as(
select C.nr_idcivil, C.nome, count(V.marca) from condutores C
inner join Veiculos_condutores VC ON C.nr_idcivil = VC.nr_idCivil
inner join Veiculos V ON  VC.matricula = V.matricula WHERE V.marca LIKE 'Toyota'
Group by c.nr_idCivil, c.nome
having count(V.marca)  IN (Select MIN(count(V.marca))  from veiculos_condutores VCD
                    INNER join veiculos VE on VCD.matricula=VE.matricula
                   WHERE VE.marca LIKE 'Toyota'
                   Group by VCD.nr_idCivil))

Select nome  from Condutores_minimo_Toyotas ;
```

Começamos por criar uma CTE (com o With).

Precisamos de dados de 3 tabelas (condutores, veiculos_condutores e veiculos) por isso fizemos os inner joins. O having count(V.marca) IN (Select MIN(count(V.marca)) vai contar a mínima ocorrência dos veículos por marca que será filtrado por marca = Toyota e selecionar os respetivos condutores.

B)

```
create view view_total_b as

select nome, NVL((select sum( cs.preco_base+(cs.custo_min *
v.duracao_minutos)+(cs.custo_km*pv.distancia_km)
+(cs.custo_espera_minuto*v.atraso_passageiro_minutos)+ (cs.preco_base+(cs.custo_min *
v.duracao_minutos)+(cs.custo_km*pv.distancia_km)
+(cs.custo_espera_minuto*v.atraso_passageiro_minutos))*cs.taxa_iva)
from  custos_servicos cs, pedidos_viagens pv, viagens v
where pv.nr_idcivil = a.nr_idcivil ), 0 ) as "custo_total_b"
from condutores a
order by "custo_total_b" desc;

select vtb.nome, vtb."custo_total_b" from view_total_b vtb
where rownum<=3;
```

Criei uma view que me dá por cada condutor a soma dos custos das suas viagens (ordenada pelo custo).

E o select que me devolve o nome dos 3 condutores (where rownum<=3;

) com maiores valor de custo.

C)

Vou criar uma view que vai ser semelhante á tabela itenerarios viagens mas vai extrair a hora atravez da data

```
create view extract_hm as
SELECT iv.cod_viagem, iv.cod_ponto_turistico, EXTRACT(HOUR FROM
    iv.hora_passagem) + EXTRACT(Minute FROM iv.hora_passagem)/60 AS hora_p
    FROM itinerarios_viagens iv;
select * from extract_hm;
```

Vou criar uma view que tem a média da hora de passagem por pontos turisticos do tipo museu

```
CREATE VIEW media_h_museus as
select avg(ehm.hora_p) as "media hora de passagem museus" from extract_hm ehm,
    pontos_turisticos pt
where ehm.cod_ponto_turistico = pt.cod_ponto_turistico and pt.tipo_ponto_turistico
    = 'MU';
select * from media_h_museus;
```

O select vai selecionar todas as viagens de viagens menos todas aquelas em que a hora de passagem, é inferior ou igual á media de hora de passagem dos museus

```
select  ehm.cod_viagem from  extract_hm ehm

minus

select ehm.cod_viagem from extract_hm ehm,  media_h_museus mhv

where ehm.hora_p <= mhv."media hora de passagem museus" ;
```

D)

```
SELECT DISTINCT

   *

FROM

   pedidos_viagens pedv

INNER JOIN viagens v ON v.cod_pedido=pedv.cod_pedido

WHERE (

      SELECT

         pedv2.data_hora_pedido

      FROM

         pedidos_viagens pedv2

      WHERE

         pedv2.matricula = pedv.matricula

         AND pedv2.data_hora_pedido < pedv.data_hora_pedido - 2/24

         AND  ROUND((v.duracao_minutos+ v.atraso_passageiro_minutos)/60,1) = 2.0

          AND pedv.cancelado LIKE 'nao'

   ) IS NOT NULL;
```

Foi feito um Select distinct para não haver resultados repetidos e um Inner Join para unir os valores da tabela pedidos viagens com os valores da tabela viagens, apos isso é chamada uma clausula Where com um sub-query, na qual seleciona novamente os pedidos de viagens e filtra, buscando os pedidos relativos a um veiculo só (pedv2.matricula = pedv.matricula) e desse veiculo, filtra-se atraves desse encherto de codigo pedv2.data_hora_pedido < pedv.data_hora_pedido - 2/24 , o ultimo pedido realizado por esse veiculo e por fim, verifica se o pedido não foi cancelado e impossibilida que valores NULL sejam retornados.

E)

Vou criar uma view com o custo total por serviço

```
create view view_total as

select cod_servico as  "Cod_servico", NVL((select sum( cs.preco_base+(cs.custo_min *
v.duracao_minutos)+(cs.custo_km*pv.distancia_km)
+(cs.custo_espera_minuto*v.atraso_passageiro_minutos)+
(cs.preco_base+(cs.custo_min * v.duracao_minutos)+(cs.custo_km*pv.distancia_km)
+(cs.custo_espera_minuto*v.atraso_passageiro_minutos))*cs.taxa_iva)
from  custos_servicos cs, pedidos_viagens pv, viagens v
where pv.cod_servico = a.cod_servico and cs.cod_servico=a.cod_servico and
v.cod_pedido=pv.cod_pedido), 0 ) as "custo_total"
from servicos a;
```

Vou criar uma view com a média do custo total por serviço

```
create view view_avg as

select cod_servico as  "Cod_servico", NVL((select avg( cs.preco_base+(cs.custo_min *
v.duracao_minutos)+(cs.custo_km*pv.distancia_km)
+(cs.custo_espera_minuto*v.atraso_passageiro_minutos)+
(cs.preco_base+(cs.custo_min * v.duracao_minutos)+(cs.custo_km*pv.distancia_km)
+(cs.custo_espera_minuto*v.atraso_passageiro_minutos))*cs.taxa_iva)
from  custos_servicos cs, pedidos_viagens pv, viagens v
where pv.cod_servico = a.cod_servico and cs.cod_servico=a.cod_servico and
v.cod_pedido=pv.cod_pedido), 0 ) as "custo_media"
from servicos a;
```

Vou criar uma view com o custo total por pedido de viagem

```
create view view_custo_pedido as

select pp.cod_pedido as  "Cod_pedido", NVL((select  sum(cs.preco_base+(cs.custo_min
* v.duracao_minutos)+(cs.custo_km*pv.distancia_km)
+(cs.custo_espera_minuto*v.atraso_passageiro_minutos)+
(cs.preco_base+(cs.custo_min * v.duracao_minutos)+(cs.custo_km*pv.distancia_km)
+(cs.custo_espera_minuto*v.atraso_passageiro_minutos))*cs.taxa_iva)
from pedidos_viagens pv , custos_servicos cs,  viagens v
```

```
where cs.cod_servico = pp.cod_servico and v.cod_pedido = pp.cod_pedido and
pv.cod_pedido=pp.cod_pedido ), 0 ) as "custo_por_pedido"
from pedidos_viagens pp;
```

Vou criar uma view que me para cada serviço quantos pedido tem um custo superior a 80% da média

```
create view m_med as
select servicos.cod_servico, count(distinct (view_custo_pedido."Cod_pedido")) as
num_de_pedidos_custo_superior_80_media  from view_custo_pedido, view_avg,
pedidos_viagens, servicos
where  view_custo_pedido."custo_por_pedido" > 1.8* view_avg."custo_media" and
pedidos_viagens.cod_pedido = view_custo_pedido."Cod_pedido" and
servicos.cod_servico = pedido_viagens.cod_servico
group by servicos.cod_servico;
```

Select apresenta o código de cada serviço, o custo de viagens global e o numero de pedidos de viagem com custo 80% superior á media

```
select m_med.*, view_total."custo_total" from m_med, view_total
where view_total."Cod_servico" = m_med.cod_servico;
```

F)

Vou criar uma view que me dá o idcivil de todos os condutores supervisores

```
create view nr_supervisor as
select distinct c.nr_idcivil_supervisor as id_supervisor from condutores c
WHERE c.nr_idcivil_supervisor IS NOT NULL;
```

Vou criar uma view que me dá o idcivil de todos os condutores não supervisores

```
create view condutor_nao_super as
select  c.nr_idcivil as num_nao_supervisor from condutores c
minus
select cc.nr_idcivil from condutores cc,  nr_supervisor ns
where cc.nr_idcivil = ns.id_supervisor;
```

Vou criar uma view que me dá o idcivil de todos os condutores supervisores e os seus pedidos de viagens

```
create view pedidos_supervisor as

select distinct ns.id_supervisor as id_superv, pv.cod_pedido as cod_ped_supervisor
from pedidos_viagens pv, veiculos_condutores vc, nr_supervisor ns

where vc.nr_idcivil = ns.id_supervisor and pv.nr_idcivil = vc.nr_idcivil;
```

Vou criar uma view com o idCivil dos supervisores e a media do custo das 20% melhores viagens

```
create view custo_melhores20pc_pedidos_por_supervisor as

select ps.id_superv as cmid_superv,  avg(vcp."custo_por_pedido")*1.2 as m_med from
pedidos_supervisor ps, view_custo_pedido vcp

where ps.cod_ped_supervisor = vcp."Cod_pedido"

group by ps.id_superv;
```

```
--view com todos os pedidos de Condutores nao supervisores—
```

Vou criar uma view com o idCivil dos condutores, e o código dos seus pedidos e o idCvivil dos seus supervisores .

```
create view pedidos_nao_supervisor as

select distinct cns.num_nao_supervisor as id_n_superv, c.nr_idcivil_supervisor as
id_super, pv.cod_pedido as cod_ped_n_supervisor from pedidos_viagens pv,
veiculos_condutores vc, condutor_nao_super cns, condutores c

where vc.nr_idcivil = cns.num_nao_supervisor and pv.nr_idcivil = vc.nr_idcivil and
c.nr_idcivil = cns.num_nao_supervisor;
```

select final que devolve o nome , nr_idcivil e numero de viagens com custo inferior a 20% da media do seu supervisor de condutores não supervisores.

```
select c.nome, pns.id_n_superv as idcivil, count(pns.cod_ped_n_supervisor) as
"numero de viagens inferior a 20% da media do supervisor"  from condutores c,
pedidos_nao_supervisor pns, view_custo_pedido vcp,
custo_melhores20pc_pedidos_por_supervisor cmps

where  cmps.cmid_superv  = pns.id_super    and vcp."Cod_pedido" =
pns.cod_ped_n_supervisor   and vcp."custo_por_pedido"< cmps.m_med and
c.nr_idcivil =  pns.id_n_superv

group by c.nome, pns.id_n_superv;
```

H)

```
Select

    c.nome, c.nr_idcivil, v.cod_viagem

from condutores c, viagens v, pedidos_viagens pv

where pv.cod_pedido = v.cod_pedido and c.nr_idcivil = pv.nr_idcivil

and not exists ( select pt.cod_ponto_turistico

from pontos_turisticos pt

minus select iv.cod_ponto_turistico

from itinerarios_viagens iv

where iv.cod_viagem = v.cod_viagem)

      and exists( select vx.* from condutores cx, viagens vx, pedidos_viagens pvx where
c.nr_idcivil = cx.nr_idcivil and pvx.nr_idcivil =cx.nr_idcivil and vx.cod_pedido
=v.cod_pedido and pvx.cod_pedido=v.cod_pedido

      and not exists( select ivy.cod_ponto_turistico, extract( hour from
ivy.hora_passagem)
```

```
from itinerarios_viagens ivy
 where ivy.cod_viagem = v.cod_viagem
 minus select ivz.cod_ponto_turistico, extract( hour from ivz.hora_passagem)
   from itinerarios_viagens ivz
where ivz.cod_viagem = vx.cod_viagem));´
```

No 1 “and not exist”

vou selecionar todos os cod_pontos_turisticos que nao existam no seu respetivo select

esse select seleciona todos os cod_pontos_turisticos que nao estejam no itenerario de cada viagem.

Ou seja o and not exist seleciona todos os cod_pontos_turisticos que estejam no itenerarios viagem de cada viagem.

No primeiro “and exists” seleciona viagens que passaram por todos os pontos turísticos.

No segundo “and not exist” seleciona todos os itenerarios que não tem uma hora diferente de passagem nas viagens, ou seja seleciona seleciona todos os itinerários que tem uma hora de passagem igual.

I)

Vou criar uma view que tem os cod_ponto_turistico dos pontos turísticos em que há mais de 5 viagens a passar por este . (a view tem também tem o numero de vezes que esse ponto turístico é passado).

```
create view pontos_tur_mais5 as;

select iv.cod_ponto_turistico as cod_ponto, count(pv.nr_idcivil) as count_nrid from
itinerarios_viagens iv, viagens v, pedidos_viagens pv

where v.cod_viagem = iv.cod_viagem and pv.cod_pedido=v.cod_pedido

group by iv.cod_ponto_turistico

having count(pv.nr_idcivil)>5;
```

Select que me devolve o nome e idCivil de condutores que fizeram viagens aos pontos turísticos que foram selecionados na view anterior

```
select distinct pv.nr_idcivil, c.nome from pedidos_viagens pv, viagens v,
itinerarios_viagens iv, pontos_tur_mais5, condutores c

where pv.cod_pedido in v.cod_pedido and v.cod_viagem in iv.cod_viagem and
iv.cod_ponto_turistico in pontos_tur_mais5.cod_ponto and c.nr_idcivil = pv.nr_idcivil;
```

Questões a Melhorar

- Oranização do codigo dos scripts .SQL
- Organização do realtorio
- Teamwork
- Comunicação
- Padrões de Desing para escrever codigos mais simples e limpos