Universidade Federal Fluminense
TCC00166 – Banco de Dados, Turma B1
P1 – 29/01/2013

Q1	
Q2	
Nota:	

Aluno: _____

Matrícula: _____ Turma: _____

1) **(6,0 pontos)** Faça o projeto de banco de dados (conceitual, lógico e físico) a situação descrita a seguir.

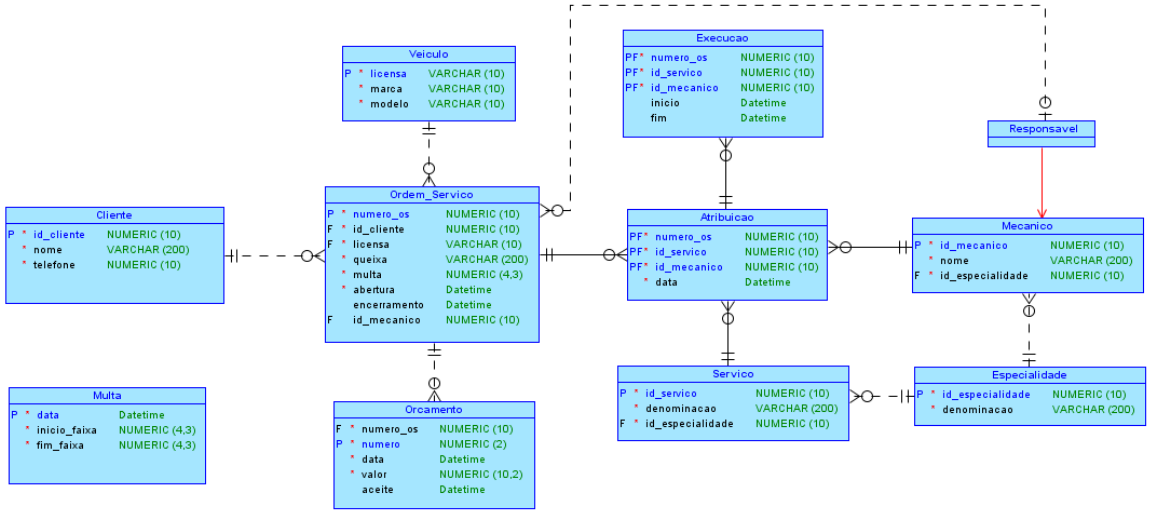
Os clientes ao chegarem à oficina registram suas queixas sobre seu veículo informando a identificação do veículo, a marca, o modelo e uma breve descrição do problema. A oficina, então, se compromete a comunicar o orçamento e prazo do reparo por telefone em até 24hs a contar da data do registro.

A partir do aceite do cliente, prazo e orçamento ficam comprometidos sendo possível uma única revisão de ambos em até 24hs a contar do momento do aceite inicial do cliente. Após o prazo de revisão do orçamento, a oficina se obriga a pagar uma multa diária por atraso correspondente a um percentual do valor do orçamento, que pode variar dentro de um intervalo estabelecido pela administração e que é negociado com o cliente pelo atendente. Os intervalos de negociação são revistos periodicamente pela administração. O valor do orçamento fica congelado e os ganhos ou perdas decorrentes de sua alteração revertem-se para a oficina.

O processo de reparo é realizado da seguinte forma. Inicialmente designa-se um mecânico responsável pelo veículo. Somente alguns mecânicos podem atuar como responsáveis e são escolhidos pela administração. Esse mecânico irá elaborar uma lista de serviços necessários. A lista é encaminhada para o coordenador da oficina que, então, distribui os serviços pelos mecânicos. Ao terminar de executar uma tarefa para a qual foram designados, os mecânicos informam o início e término do serviço. Os mecânicos responsáveis podem também ser designados pelo coordenador da oficina. A qualquer tempo o mecânico responsável pode reabrir uma tarefa dada como encerrada por um mecânico caso avalie que o serviço não foi concluído a contento. A quantidade de divergências entre o responsável e o mecânico que executou a tarefa é utilizada como medida de qualidade. A qualquer tempo, também, o mecânico responsável pode incluir novos serviços necessários. Qualquer serviço em aberto pode ser designado a outro mecânico sendo que o mecânico anterior deverá registrar o tempo parcial que utilizou na realização da tarefa e, se necessário, comentários sobre o que falta a ser feito. Ao final de todos os serviços o mecânico responsável encerra o reparo. Ao final de cada dia a oficina entra em contato com os clientes dos reparos atrasados para reagendar a entrega. Os mecânicos são organizados em por área de conhecimento. Existem, eletricistas, lanterneiros e mecânicos.

Obs.: 1) Alguns atributos estão implícitos e devem ser propostos, 1) lembre-se de escrever em lista separada as regras de integridade que sejam necessárias e 2) as regras devem ser implementadas no projeto físico.

R.:



Regras:

i. Multa combinada com o cliente deve ser permitida
ii. Abertura de ordem de serviço deve ser anterior a todos os orçamentos
iii. Encerramento da ordem de serviço deve ser posterior a todos os orçamentos e execuções de atribuições
iv. Só pode haver no máximo dois orçamentos para cada ordem de serviço
v. As atribuições e execuções devem ser posteriores ao aceite definitivo do cliente
vi. Especialidade do serviço deve ser compatível com especialidade do mecânico.

```
CREATE TABLE Atribuicao (
   numero_os NUMBER (10)  NOT NULL ,
   id_servico NUMBER (10)  NOT NULL ,
   ID_MECANICO NUMBER (10)  NOT NULL ,
   DATA DATE  NOT NULL,
   PRIMARY KEY ( NUMERO_OS, ID_SERVICO, ID_MECANICO ),
   CONSTRAINT E_COMPOSTA_DE FOREIGN KEY (NUMERO_OS) REFERENCES ORDEM_SERVICO
(NUMERO_OS),
   CONSTRAINT EXECUTA FOREIGN KEY (ID_MECANICO) REFERENCES MECANICO (ID_MECANICO),
   CONSTRAINT RECEBE FOREIGN KEY (ID_SERVICO) REFERENCES Servico (id_servico));

CREATE TABLE Cliente (
   id_cliente NUMBER (10)  NOT NULL ,
   nome VARCHAR2 (200)  NOT NULL ,
   TELEFONE NUMBER (10)  NOT NULL ,
   PRIMARY KEY ( id_cliente ) );


CREATE TABLE Especialidade (
   id_especialidade NUMBER (10)  NOT NULL ,
   DENOMINACAO VARCHAR2 (200)  NOT NULL ,
   PRIMARY KEY ( id_especialidade ) );
```

```
CREATE TABLE Execucao  (
    numero_os NUMBER (10)  NOT NULL ,
    id_servico NUMBER (10)  NOT NULL ,
    id_mecanico NUMBER (10)  NOT NULL ,
    inicio DATE ,
    FIM DATE ,
    PRIMARY KEY ( NUMERO_OS, ID_SERVICO, ID_MECANICO ),
    CONSTRAINT EXECUCAO_DATAS_CK CHECK (FIM > INICIO OR FIM IS NULL),
    CONSTRAINT  GERA  FOREIGN  KEY  (NUMERO_OS,ID_SERVICO,ID_MECANICO)  REFERENCES  Atribuicao
(numero_os,id_servico,id_mecanico));


CREATE TABLE Mecanico  (
    id_mecanico NUMBER (10)  NOT NULL ,
    nome VARCHAR2 (200)  NOT NULL ,
    ID_ESPECIALIDADE NUMBER (10)  NOT NULL ,
    PRIMARY KEY ( ID_MECANICO ),
    CONSTRAINT  AGRUPA_MECANICO  FOREIGN  KEY  (ID_ESPECIALIDADE)  REFERENCES  Especialidade
(id_especialidade));


CREATE TABLE Multa  (
    data DATE  NOT NULL ,
    inicio_faixa NUMBER (4,3)  NOT NULL ,
    FIM_FAIXA NUMBER (4,3)  NOT NULL ,
    PRIMARY KEY ( data ) );


CREATE TABLE Orcamento  (
    numero_os NUMBER (10)  NOT NULL ,
    numero NUMBER (2)  NOT NULL ,
    data DATE  NOT NULL ,
    valor NUMBER (10,2)  NOT NULL ,
    ACEITE DATE  ,
    PRIMARY KEY ( NUMERO, NUMERO_OS ) ,
    CONSTRAINT ORCAMENTO_DATAS_CK CHECK (ACEITE > "DATA" OR ACEITE IS NULL),
    CONSTRAINT POSSUI FOREIGN KEY (NUMERO_OS) REFERENCES Ordem_Servico (numero_os));


CREATE TABLE Ordem_Servico  (
    numero_os NUMBER (10)  NOT NULL ,
    id_cliente NUMBER (10)  NOT NULL ,
    licensa VARCHAR2 (10)  NOT NULL ,
    queixa VARCHAR2 (200)  NOT NULL ,
    multa NUMBER (4,3)  NOT NULL ,
    abertura DATE  NOT NULL ,
    ENCERRAMENTO DATE ,
    ID_MECANICO NUMBER (10) ,
    PRIMARY KEY ( numero_os ),
    CONSTRAINT ABRE FOREIGN KEY (ID_CLIENTE) REFERENCES CLIENTE (ID_CLIENTE),
```

```
    CONSTRAINT CUIDA FOREIGN KEY (ID_MECANICO) REFERENCES MECANICO (ID_MECANICO),
    CONSTRAINT SOFRE FOREIGN KEY (LICENSA) REFERENCES Veiculo (licensa));


CREATE TABLE Servico     (
    id_servico NUMBER (10)  NOT NULL ,
    denominacao VARCHAR2 (200)  NOT NULL ,
    ID_ESPECIALIDADE NUMBER (10)  NOT NULL),
    PRIMARY KEY ( id_servico ),
    CONSTRAINT    AGRUPA_SERVICO    FOREIGN    KEY    (ID_ESPECIALIDADE)    REFERENCES    Especialidade
(id_especialidade));


CREATE TABLE Veiculo     (
    licensa VARCHAR2 (10)  NOT NULL ,
    marca VARCHAR2 (10)  NOT NULL ,
    MODELO VARCHAR2 (10)  NOT NULL ,
    PRIMARY KEY ( licensa ) );

create or replace TRIGGER VALIDA_ORDEM_SERVICO BEFORE INSERT OR UPDATE ON "ORDEM_SERVICO"
REFERENCING OLD AS ANTIGO NEW AS NOVO for each row
DECLARE
 INICIO NUMBER(4,3); FIM NUMBER(4,3); "DATA" date;
BEGIN

  -- Verifica se o valor da multa é permitido
  WITH
    T1 AS (SELECT MAX("DATA") AS "VIGENCIA" FROM MULTA WHERE "DATA"<=:NOVO.ABERTURA)
            SELECT INICIO_FAIXA,FIM_FAIXA INTO INICIO,FIM FROM "MULTA" WHERE "DATA"=
           (SELECT "VIGENCIA" FROM T1);
  if (not(:NOVO.MULTA between INICIO and FIM)) then
    RAISE_APPLICATION_ERROR(-20001,'ERRO');
  END IF;

  -- Verifica data abertura
  SELECT MIN("DATA") INTO "DATA" FROM "ORCAMENTO" WHERE "NUMERO_OS"=:NOVO."NUMERO_OS";
  IF (:NOVO.ABERTURA > "DATA") THEN
    RAISE_APPLICATION_ERROR(-20001,'ERRO');
  END IF;

 -- Verifica encerramento
  WITH
    T1 AS (SELECT MAX(T2."FIM") AS "DATA" FROM ATRIBUICAO T1 LEFT JOIN EXECUCAO T2 ON
            T1.NUMERO_OS = T2.NUMERO_OS AND T1.ID_SERVICO=T2.ID_SERVICO AND T1.ID_MECANICO =
            T2.ID_MECANICO
            WHERE T1.NUMERO_OS=:NOVO.NUMERO_OS),
    T2 AS (SELECT MAX(T2."INICIO") AS "DATA" FROM ATRIBUICAO T1 LEFT JOIN EXECUCAO T2 ON
            T1.NUMERO_OS = T2.NUMERO_OS AND T1.ID_SERVICO=T2.ID_SERVICO AND T1.ID_MECANICO =
            T2.ID_MECANICO
            WHERE T1.NUMERO_OS=:NOVO.NUMERO_OS),
    T3 AS (SELECT MAX("DATA") AS "DATA" FROM ATRIBUICAO WHERE NUMERO_OS=:NOVO.NUMERO_OS),
    T4 AS (SELECT MAX("ACEITE") AS "DATA" FROM "ORCAMENTO" WHERE "NUMERO_OS" =
```

```
            :NOVO."NUMERO_OS"),
   T5 AS (SELECT "DATA" FROM T1 UNION SELECT "DATA" FROM T2 UNION SELECT "DATA" FROM T3
            UNION SELECT "DATA" FROM T4)
  SELECT MAX("DATA") INTO "DATA" FROM T5;

  IF (:NOVO.ENCERRAMENTO < "DATA") THEN
    RAISE_APPLICATION_ERROR(-20001,'ERRO');
  END IF;
END;

CREATE OR REPLACE
TRIGGER VALIDA_ORCAMENTO
BEFORE INSERT OR UPDATE ON "ORCAMENTO"
REFERENCING OLD AS ANTIGO NEW AS NOVO
for each row
DECLARE
  CARDINALIDADE NUMBER(2);
BEGIN

  -- Verifica cardinalidade
  SELECT COUNT(*) INTO CARDINALIDADE FROM ORCAMENTO WHERE NUMERO_OS=:NOVO.NUMERO_OS;
  IF ((:NOVO.NUMERO_OS != :ANTIGO.NUMERO_OS or :ANTIGO.NUMERO_OS IS NULL) AND
      CARDINALIDADE =2) THEN
    RAISE_APPLICATION_ERROR(-20001,'ERRO');
  END IF;
  -- Verificar data
END;

create or replace
TRIGGER VALIDA_ATRIBUICAO
BEFORE INSERT OR UPDATE ON "ATRIBUICAO"
REFERENCING OLD AS ANTIGO NEW AS NOVO
for each row
DECLARE
  ID1 NUMBER(10);
  ID2 NUMBER(10);
BEGIN

  -- Verifica especialidade
  SELECT ID_ESPECIALIDADE INTO ID1 FROM MECANICO WHERE ID_MECANICO=:NOVO.ID_MECANICO;
  SELECT ID_ESPECIALIDADE INTO ID2 FROM SERVICO WHERE ID_SERVICO=:NOVO.ID_SERVICO;

  IF (ID1 != ID2) THEN
    RAISE_APPLICATION_ERROR(-20001,'ERRO');
  END IF;
END;
```

2) **(4,0 pontos)** Escreva em álgebra relacional e depois em SQL as consultas abaixo.

Empregado (cpf,nome,habilidade,gerente)

Projeto (código,orçamento)
Habilidade (código,denomicação,projeto)
Trabalha_em (projeto,empregado)

a) Liste o CPF e o nome dos empregados que são contadores e trabalham em projetos com orçamento superior a R$100.000,00.
b) Liste o nome de cada empregado e o número de empregados do qual ele é gerente.
c) Liste o código de cada projeto que necessita de uma habilidade que nenhum dos empregados que nele trabalha possui.