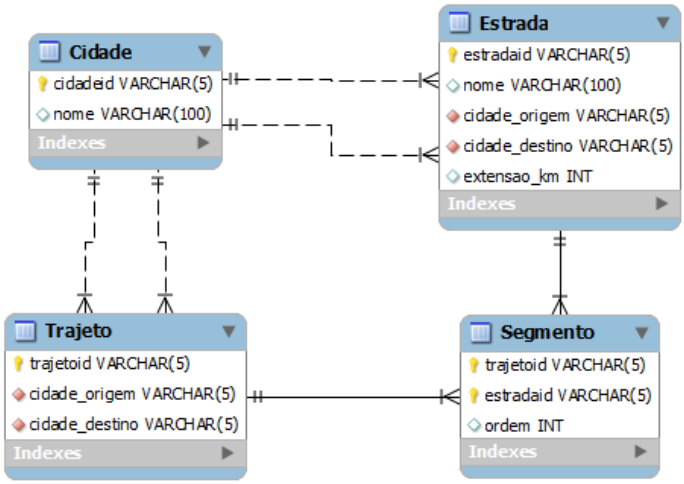
Lista de Exercícios (respostas)

Bancos de Dados Instituto de Computação Universidade Estadual de Campinas		Query e Normalização 2016 André Santanchè

Considere o diagrama abaixo representa graficamente um modelo relacional de tabelas que controlam cidades, estradas e trajetos entre cidades. A tabela de Cidades mantém um cadastro de cidades; a tabela de Estradas registra estradas que ligam uma cidade (cidade_origem) a outra (cidade_destino), bem como sua quilometragem. Cada registro da tabela Trajeto especifica um trajeto, que consiste em uma sequência ordenada de estradas que ligam duas cidades (cidade_origem e cidade_destino), por exemplo, um trajeto entre Salvador e Curitiba, pode envolver uma sequência de estradas: Salvador-Belo Horizonte, Belo Horizonte-São Paulo e São Paulo-Curitiba. A tabela Segmento associa estradas a trajetos. O campo ordem é um campo numérico sequencial (iniciado de 1 para cada trajeto) usado para ordenar os segmentos (estradas) dentro de um trajeto.



Questão 1

A partir do esquema apresentado, escreva as seguintes sentenças SQL:

```
-- Relacionando as três tabelas: Trajeto, Segmento e Estrada

SELECT T.trajetoid, T.cidade_origem, T.cidade_destino, S.ordem, S.estradaid, E.cidade_origem,
E.cidade_destino, E.extensao_km
 FROM Trajeto T, Segmento S, Estrada E
 WHERE T.trajetoid = S.trajetoid AND S.estradaid = E.estradaid
 ORDER BY T.trajetoid, S.ordem;
```

a) Calcule a quilometragem total de cada Trajeto.

```
SELECT T.trajetoid, SUM(E.extensao_km)
 FROM Trajeto T, Segmento S, Estrada E
 WHERE T.trajetoid = S.trajetoid AND S.estradaid = E.estradaid
 GROUP BY T.trajetoid;
```

b) Para que um trajeto seja consistente, a cidade de origem deste trajeto deve ser igual à cidade de origem cadastrada no primeiro segmento do respectivo trajeto. Escreva uma consulta SQL que mostre a identificação dos trajetos que não estão consistentes conforme este critério.

```
-- (passo 1) Primeiro segmento de cada trajeto
SELECT * FROM Segmento WHERE ordem=1;
```

```
-- (passo 2) Associando o primeiro segmento de cada Trajeto
SELECT *
  FROM Estrada, Segmento, Trajeto
  WHERE Estrada.estradaid = Segmento.estradaid AND Segmento.trajetoid = Trajeto.trajetoid AND
      Segmento.ordem = 1

-- (passo final)
SELECT T.trajetoid
  FROM Trajeto T, Segmento S, Estrada E
  WHERE T.trajetoid = S.trajetoid AND S.estradaid = E.estradaid AND
      T.cidade_origem <> E.cidade_origem AND S.ordem = 1;
```

c) Outro critério para que um trajeto seja consistente é que a cidade de destino deste trajeto deve ser igual à cidade de destino cadastrada no último segmento do respectivo trajeto. Escreva uma consulta SQL que mostre a identificação dos trajetos que não estão consistentes conforme este critério.

```
-- * (abordagem 1) Usando VIEW
-- (passo 1a) Criando uma VIEW que computa a ordem da Estrada final de cada Trajeto (com MAX)
CREATE VIEW TrajetoNumSeg AS
SELECT S.trajetoid trajid, MAX(S.ordem) numseg
 FROM Segmento S
 GROUP BY S.trajetoid

-- (passo 1b) Criando uma VIEW que computa a ordem do Estrada final de cada Trajeto (com COUNT)
CREATE VIEW TrajetoNumSeg AS
SELECT S.trajetoid trajid, COUNT(*) numseg
 FROM Segmento S
 GROUP BY S.trajetoid;

-- (passo 2a) Criando a VIEW que define a ordem da estrada final de cada Trajeto e sua Cidade
Destino (com MAX)
CREATE VIEW TrajetoNumSeg AS
SELECT T.trajetoid trajid, T.cidade_destino destino, MAX(S.ordem) numseg
 FROM Trajeto T, Segmento S
 WHERE T.trajetoid = S.trajetoid
 GROUP BY T.trajetoid;

-- (passo 2b) Criando a VIEW que define a ordem da estrada final de cada Trajeto e sua Cidade
Destino (com COUNT)
CREATE VIEW TrajetoNumSeg AS
SELECT T.trajetoid trajid, T.cidade_destino destino, COUNT(*) numseg
 FROM Trajeto T, Segmento S
 WHERE T.trajetoid = S.trajetoid
 GROUP BY T.trajetoid;

-- (passo 3) Verificando o resultado da VIEW
SELECT * FROM TrajetoNumSeg;
```

```
-- (passo final) Relacionando a VIEW com Segmento e Estrada
SELECT T.trajid
  FROM TrajetoNumSeg T, Segmento S, Estrada E
  WHERE T.trajid = S.trajetoid AND S.estradaid = E.estradaid AND
        T.destino <> E.cidade_destino AND S.ordem = T.numseg;

-- como se desfazer da VIEW depois do uso
DROP Table TrajetoNumSeg;

-- * (abordagem 2) Usando SELECT aninhado
-- passo 1
SELECT T.trajetoid
  FROM Trajeto T, Segmento S1, Estrada E
  WHERE E.estradaid = S1.estradaid AND S1.trajetoid = T.trajetoid
        AND S1.ordem = (
                SELECT MAX(ordem) FROM Segmento S2 WHERE S2.trajetoid = S1.trajetoid)
                AND E.cidade_destino <> T.cidade_destino

-- passo 2
SELECT T.trajetoid, T.cidade_origem, T.cidade_destino, SUM(E.extensao_km)
  FROM Trajeto T, Segmento S, Estrada E
  WHERE T.trajetoid = S.trajetoid AND S.estradaid = E.estradaid
  GROUP BY T.trajetoid, T.cidade_origem, T.cidade_destino;
```

d) Retorne os trajetos de menor quilometragem entre cada origem/destino diferente. Esta consulta deve apresentar para cada um dos trajetos selecionados: cidade origem, cidade destino, menor quilometragem entre elas. A quilometragem de cada trajeto é calculada pela soma da quilometragem de todas as estradas que compõem o trajeto.

```
CREATE VIEW TotalTrajeto AS
SELECT  T.trajetoid trajid, T.cidade_origem origem, T.cidade_destino destino, SUM(E.extensao_km)
        extensao
  FROM Trajeto T, Segmento S, Estrada E
  WHERE T.trajetoid = S.trajetoid AND S.estradaid = E.estradaid
  GROUP BY T.trajetoid, T.cidade_origem, T.cidade_destino;
-----------------------------------------------------------------------------------------------------
SELECT origem, destino, MIN(extensao)
  FROM TotalTrajeto
  GROUP BY origem, destino;

-----------------------------------------------------------------------------------------------------
DROP TABLE TotalTrajeto;
```

e) Considerando que foi acrescentado na tabela de Trajetos um campo `extensao_km` que contém a quilometragem total do trajeto, escreva uma sentença que calcule e atualize o campo `extensao_km` da tabela de Trajetos, baseando-se na soma da quilometragem das estradas.

```
UPDATE Trajeto T
  SET T.extensao_km =
     (SELECT SUM(E.extensao_km)
        FROM Segmento S, Estrada E
        WHERE T.trajetoid = S.trajetoid AND
            S.estradaid = E.estradaid);
```

f) Retorne o nome das cidades que não aparecem na origem de nenhum segmento (questão de prova).

Questão 2 (questões de prova)

Considere os comandos SQL a seguir para criar tabelas que controlam Produtos e Receitas, bem como o respectivo esquema relacional simplificado. A tabela de `Produto` mantém um cadastro de produtos, com seu `código`, `nome` e `custo_unitario` que corresponde ao custo de aquisição de uma unidade do produto. Cada `Receita` tem um `código` e `nome`. Cada entrada nesta tabela `Ingrediente` indica que um `Produto` é componente de uma `Receita` em uma certa `quantidade`.

```
CREATE TABLE Produto (
  codigo_produto VARCHAR(5),
  nome_produto VARCHAR(80),
  custo_unitario FLOAT,
  PRIMARY KEY (codigo_produto)
);
```

```
CREATE TABLE Ingrediente (
  codigo_receita VARCHAR(5),
  codigo_produto VARCHAR(5),
  quantidade FLOAT,
  PRIMARY KEY (codigo_receita, codigo_produto, quantidade),
  FOREIGN KEY (codigo_receita)
    REFERENCES Receita (codigo_receita),
  FOREIGN KEY (codigo_produto)
    REFERENCES Produto (codigo_produto)  );
```

```
CREATE TABLE Receita (
  codigo_receita VARCHAR(5),
  nome_receita VARCHAR(80),
  custo_total FLOAT,
  PRIMARY KEY (codigo_receita)  );
```

Esquema Relacional:
Produto(codigo_produto, nome_produto, custo_unitario)
Receita(codigo_receita, nome_receita, custo_total)
Ingrediente(codigo_receita, codigo_produto, quantidade)

A partir do esquema apresentado, escreva as seguintes consultas SQL:

a) Escreva uma consulta que liste o nome dos `Produtos` que aparecem em mais de uma `Receita`.
b) Em algumas receitas o mesmo produto aparece mais de uma vez com quantidades diferentes. Crie uma nova tabela de Ingredientes a partir de uma View em que não haja produtos que aparecem mais de uma vez. Para isso, junte os produtos que aparecem mais de uma vez na mesma receita e some as suas quantidades.
c) Estenda a questão da letra (b) aplicando a seguinte regra: se o produto aparecer duas vezes, junte os dois em um e some as suas quantidades, se aparecer mais do que duas vezes ele não deve entrar na tabela Ingredientes nova.
d) Escreva uma consulta que mostre o nome das receitas que não têm produtos que aparecem mais de uma vez.
e) Escreva uma consulta que liste o nome daqueles `Produtos` que não aparecem em nenhuma `Receita`.
f) O `custo_total` de uma `Receita` é calculado pelo somatório do `custo` de cada ingrediente multiplicado pela sua `quantidade` na receita. Escreva uma consulta que apresente o nome de todas as receitas cujo `custo_total` não atende a este critério.
g) Escreva uma cláusula de UPDATE que calcule e atualize o campo `custo_total` da receita a partir dos ingredientes, conforme indicado em (f).