Fundação CECIERJ - **Vice Presidência de Educação Superior a Distância**

Curso de Tecnologia em Sistemas de Computação
Disciplina: Programação II
Gabarito da AD2 - 2° semestre de 2014

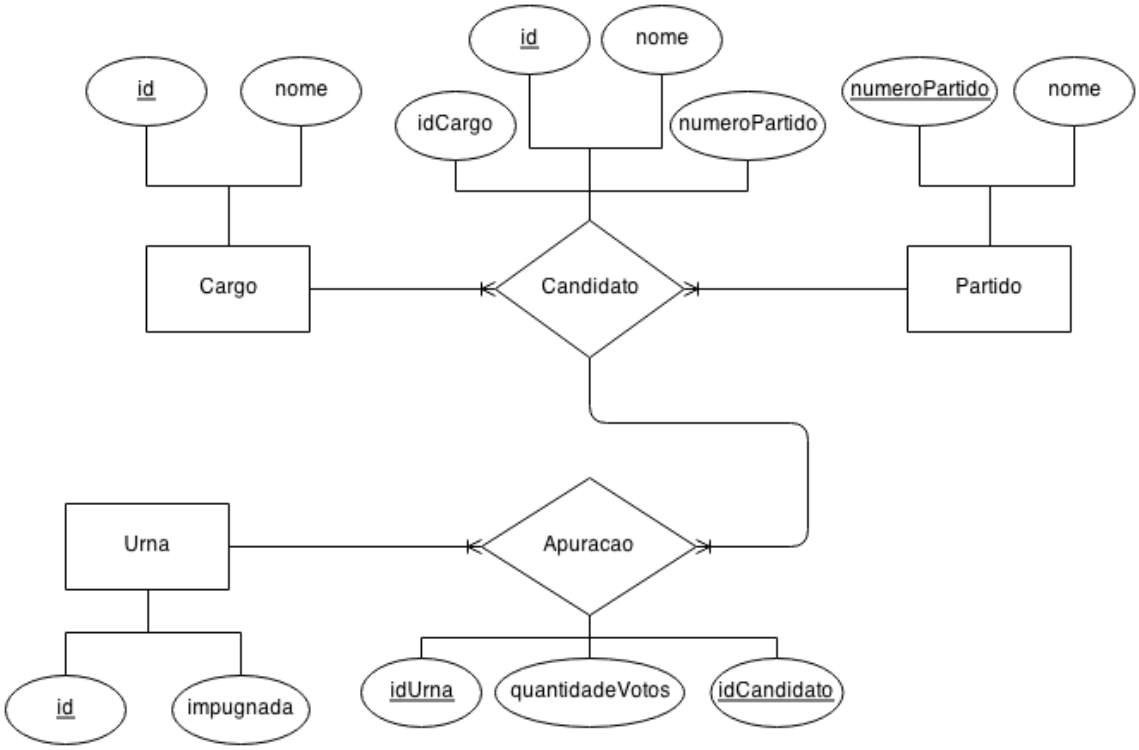
Deseja-se construir um sistema para administrar eleições majoritárias, isto é, eleições onde cada cargo é disputado por apenas um candidato de cada partido. O sistema deve contemplar os seguintes casos de uso:

I. *Registro de cargos a serem disputados*: os cargos a serem disputados na eleição, por exemplo, "Presidente", "Governador", "Prefeito", etc, são entrados no banco de dados.
II. *Registro de partidos*: cada partido político que deseja registrar candidatos às eleições é registrado no banco de dados com um nome e um número.
III. *Registro de candidatos*: cada candidato que disputará as eleições registra seu nome, partido e número. Apenas um candidato por partido é admitido.
IV. *Apuração de urna*: Para cada urna aberta, o número de votos de cada candidato é inserido no banco de dados. É possível também registrar votos de legenda, isto é, votos em que o eleitor manifesta preferência por um partido, o que é considerado pelo sistema como um voto em cada candidato daquele partido.
V. *Cancelamento de urna*: Os dados de cada urna devem poder ser retirados do sistema em caso de recontagem de votos ou impugnação. Se uma urna é cancelada, todos os votos daquela urna são retirados do banco de dados.
VI. *Proclamação do resultado das eleições*. Para cada cargo, o candidato que recebe o maior número de votos vence. Em caso de dois ou mais candidatos receberem o mesmo número de votos, é anunciado um empate.

Pede-se:

1. (2 pontos) Faça um diagrama de entidades e relacionamentos para o banco de dados.

R:



Aproveitando que a relação numeroPartido é única para os diferentes candidatos a cada cargo de um mesmo partido, o candidato pode usar o numero de sua legenda como seu número e como sua chave para partido. Isso é útil nas consultas, como veremos.

2. (1 ponto) Escreva uma modelagem física em SQL para o banco de dados.

R:

```
CREATE TABLE `cargo` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `nome` varchar(255) NOT NULL,
  PRIMARY KEY (`id`)
);


CREATE TABLE `partido` (
  `numeroPartido` int(11) NOT NULL,
  `nome` varchar(255) NOT NULL,
  PRIMARY KEY (`numeroPartido`)
);
```

```
CREATE TABLE `urna` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `impugnada` bit(1) NOT NULL DEFAULT FALSE,
  PRIMARY KEY (`id`)
);

CREATE TABLE `candidato` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `numeroPartido` int(11) NOT NULL,
  `idCargo` int(11) NOT NULL,
  `nome` varchar(255) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `UQ_candidato_partido_cargo`
(`numeroPartido`,`idCargo`),
  FOREIGN KEY (`idCargo`) REFERENCES `cargo` (`id`),
  FOREIGN KEY (`numeroPartido`) REFERENCES `partido`
(`numeroPartido`)
);

CREATE TABLE `apuracao` (
  `idUrna` int(11) NOT NULL,
  `idCandidato` int(11) NOT NULL,
  `numeroVotos` int(11) NOT NULL,
  PRIMARY KEY (`idUrna`,`idCandidato`),
  FOREIGN KEY (`idCandidato`) REFERENCES `candidato`
(`id`),
  FOREIGN KEY (`idUrna`) REFERENCES `urna` (`id`)
);
```

3. (2 pontos) Escreva em SQL consultas para obter as seguintes informações:
   a) Quais candidatos ao cargo "Prefeito" não receberam votos?

```
SELECT c.nome
  FROM Candidato c,
       (  SELECT SUM(a.numeroVotos) as qtdVotos,
                 c.id as idCandidato
            FROM Apuracao a
      INNER JOIN Candidato c ON c.id = a.idCandidato
      INNER JOIN Cargo cg ON cg.id = c.idCargo
      WHERE cg.nome = 'Prefeito') subQuery
 WHERE  c.id = subQuery.idCandidato
   AND  subQuery.qtdVotos = 0;
```

   b) Qual o partido que recebeu mais votos para o cargo "Governador"?

```
R:   SELECT    SUM(numeroVotos) qtd,
               numeroPartido
       FROM    Apuracao
 INNER JOIN Candidato ON Apuracao.idCandidato =
                         Candidato.id
   GROUP BY idCandidato ORDER BY qtd DESC LIMIT 1;
```

4. (3 pontos) Usando PHP, HTML e MySQL, escreva o código para uma página web para exibir e eventualmente modificar todos os cargos sendo disputados na eleição (caso de uso I). Você pode assumir que não mais do que cinco cargos podem ser disputados por eleição. Assim a página deve exibir 5 campos de texto editáveis e um botão "Submeter" que, se apertado, inclui ou substitui no banco de dados os cargos a serem disputados.

R; Espera-se que o aluno escreva pelo menos 1 arquivo PHP onde posicione um formulário com método GET ou POST, em que hajam 5 campos input e um botão do tipo submit.

No lado do PHP, o aluno recupere no array do respectivo método os 5 valores, e adote alguma das estratégias a seguir:

- **Apagar todos, inserir todos:** é a estratégia mais ingênua, mas mais eficiente. Trata-se de chamar um DELETE * FROM Cargo e em seguida, a chamada de 1 a 5

INSERTs na tabela Cargo. É preciso verificar no loop se o nome do cargo está nulo e dar break, Essa resposta será aceita como as outras.

- **Manter os já existentes, apagar removidos, inserir novos:** é uma estratégia mais elaborada, mas não há ganho real para o problema da questão. Envolve um SELECT para os cargos já presentes, e uma operação condicionada com a existência ou não do cargo
- **Gerenciar instâncias de eleições:** uma estratégia mais elaborada do que a proposta pelo gabarito é a de criar mais uma entidade, Eleicao, a qual se atribuiria 5 cargos "hard_coded": id_cargo_1, id_cargo_2, etc. Apenas o primeiro cargo seria obrigatório, permitindo valores null nos demais. Dessa forma, os registros da tabela Cargo seriam reaproveitáveis entre eleições. Importante que o aluno não atualize a tabela Cargo quando ocorre a mudança de nome, mas crie novas instâncias de Cargo, e atribua a elas a eleição atual. Após isso, remover as istâncias de Cargo as quais nenhuma Eleicao referencie. Esta seria a melhor implementação e o tutor pode dar décimos extras se bem implementada.

5. (2 pontos) Usando PHP, HTML e MySQL, escreva o código para uma página web que exibe o resultado da eleição (caso de uso VI).

R: Na modelagem escolhida pelo gabarito, votos de legenda nada mais são do que um voto simultâneo para todos os candidatos a cargos daquele partido. O tratamento dos votos de legenda se dá neste caso, no momento da inserção, onde o sistema identificaria que se trata de um voto de legenda, e adicionaria tantos inserts quanto fossem os cargos para o partido de mesmo número.

Mas há o caso em que o aluno considere um campo para indicar votos de legenda, para isso, seria necessário contabilizar a apuração voto a voto, utilizando um modelo para indicar o Eleitor, que estaria amarrado em Urna. Seus votos poderiam estar "hard_coded" na tabela Eleitor (voto_cargo_1, voto_cargo_2, etc.) ou via uma tabela de relacionamento Voto, que faria a ponte entre Eleitor e Candidato. A tabela Apuracao deixaria de existir e suas informações seriam calculadas dinamicamente pelos relacionamentos entre Voto e Urna, considerando também os casos de voto único para legenda, que devem ser multiplicados durante a contagem.

Em função das escolhas do gabarito, a query principal responsável por atender este caso:

```
SELECT  CASE Urna.impugnada
              WHEN TRUE THEN 0
              ELSE SUM(numeroVotos)
        END as qtd,
        numeroPartido
  FROM  Apuracao
```

```
INNER JOIN     Candidato ON Apuracao.idCandidato =
Candidato.id

INNER JOIN     Urna ON Apuracao.idUrna = Urna.id

  GROUP BY idCandidato ORDER BY qtd DESC;
```