**UNIVERSIDADE FEDERAL DOS VALES DO JEQUITINHONHA E MUCURI**
**FACULDADE DE CIÊNCIAS EXATAS E TECNOLÓGICAS - FACET**
**DEPARTAMENTO DE COMPUTAÇÃO - DECOM**
**SISTEMAS DE INFORMAÇÃO**
**DIAMANTINA- MG**

# Trabalho Prático

**Docente:** Elias Alves
**Disciplina:** Bancos de Dados II
**Tema:** Trabalho de Banco de Dados II
**Data de Entrega:** 28/01/2017
**Valor:** 25 pontos
**Instruções iniciais:**

- A atividade será desenvolvida em grupo, sendo obrigatória a contribuição direta de todos os participantes no envio de códigos ou relatórios via GitHub. O envio do trabalho será feito via repositório específico criado pela equipe. No dia da entrega haverá também uma pequena entrevista com o grupo.
- Para a implementação você precisará de um servidor *PostgreSQL*, além de um cliente com acesso ao servidor. Recomenda-se o uso da versão 9.6 do *PostgreSQL*, bem como o *pgAdmin*, como cliente, mas fica a cargo do estudante a escolha do ambiente. Você deverá usar a mesma base de dados das listas de exercício prático.
- O trabalho se constituirá de um conjunto de consultas a serem construídas e entregues em formato eletrônico sobre as bases de dados concurso, banco e companhia. Os alunos serão divididos em 4 grupos de 3 alunos e 2 participações individuais, observando a mesma composição dos temas para seminários:
  1. NoSQL: Rafael, Cristian, Lucas Moreno
  2. BDs Orientados a Objetos:Wallace, Mariana, Diego
  3. XML/Json: Breno, Réggis, Ana
  4. BDs Semânticos: Lucas Porto, Amanda, Mayko
  5. BDs Multimídia: Pedro
  6. BDs Grográficos: Caio
- Cada grupo ficará responsável pela elaboração de 6 exercícios. Os que farão sozinhos poderão eliminar um exercício entre os sorteados, ou seja, apresentarão 5. A distribuição das questões para os grupos foi feita por sorteio[1] e é a seguinte:
  - Grupo 1: Questões: 1 - 3 - 4 - 5 - 7 - 10
  - Grupo 2: Questões: 2 - 4 - 6 - 7 - 8 - 10
  - Grupo 3: Questões: 1 - 3 - 4 - 5 - 9 - 10
  - Grupo 4: Questões: 1 - 3 - 4 - 7 - 8 - 9
  - Grupo 5: Questões: 2 - 4 - 6 - 7 - 9 - 10

[1] https://sorteador.com.br/?de=1&ate=10&animacao=0&quantidade=6&ordenar=1

- Serão disponibilizados códigos-fonte de templates em Latex e Open Document Format (*.odt) para apresentação dos relatórios, porém seu uso não é obrigatório.
- O último commit com as alterações deverá ser realizado até as 23h59min da data definida, mas os outros membros da equipe já deverão ter enviado suas contribuições.
- As dúvidas deverão ser encaminhadas ao canal principal do **Slack** para que todos os membros possam participar das discussões.

# Questões a serem desenvolvidas

1. Criar uma função que recebe como parâmetro a matricula do funcionário, calcula e retorna a sua idade, em anos. (DICA: utilize as funções de manipulação de valores do tipo data).
2. Crie uma **view** que selecione a matricula do funcionário, o nome do funcionário, a sua idade, o nome de seu gerente e a matricula de seu gerente.
3. Criar uma **função** que recebe como parâmetro a data de nascimento do funcionário e retorna o dia da semana (em português) em que ele nasceu.
4. Criar uma **view** que mostra, para cada funcionário, seus dados e o nome do departamento onde ele trabalha.
5. Criar uma **stored procedure** que receba como parâmetros: o código de um cliente e um valor de saldo e mostre os dados das contas daquele cliente cujo saldo é maior que o valor dado como parâmetro.
6. Criar uma **view** que mostre, para cada dependente, seus dados pessoais, seu parentesco com o funcionário e o nome e endereço do funcionário.
7. Crie um modelo simplificado do banco de dados '**banco**', com as seguintes informações: a tabela *conta* terá um campo que armazenará o saldo inicial da conta e o saldo atual.
   (a) O saldo atual deverá ser atualizado por uma **trigger**, que será automaticamente executada sempre que uma operação de débito ou crédito for feita. (crie uma nova tabela para registrar as operações de crédito e débito nas contas). Obs.: As demais tabelas do banco original não precisam ser criadas.
8. Dado o banco de dados teste com as tabelas t1(nome, nascimento, salario) e t2(codigo, nome, rua, bairro), escreva os seguintes comandos:
   (a) comando para criação de um usuário chamado *'user1'* que terá totais privilégios apenas para o banco de dados teste. A senha será *'suser1'*. O usuário não terá privilégios de **grant**.
   (b) Retire os privilégios de update e insert de *'user1'* para os campos da tabela t1

   (c) Retire o privilégio de delete para o campo bairro da tabela t2

   (d) Conceda todos os privilégios de *'user1'* para todos os campos de t2

   (e) Retire todos os privilégios de *'user1'* para todos os campos de t2

9. Crie um novo super-usuário chamado *'gerente'* com senha igual a *'root'*, com opção de **grant**.

   (a) Retire o privilégio de grant do superusuário criado anteriormente.

   (b) Exiba os privilégios do usuário *'user1'*.

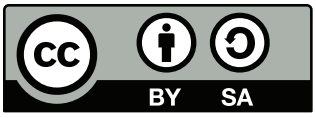
   (c) Altere o arquivo *pg_hba.conf* para que o usuário gerente possa se conectar somente de um host com ip dentro do intervalo de ip's da rede do servidor[2][3].

10. Criar uma stored procedure que retorna o nome dos clientes que possuem conta em alguma agência que Jones também possui conta e que não possuem empréstimo em agências onde Smith possui empréstimo. (obs.: O nome do cliente Jones não deve aparecer no resultado).

**Forma de avaliação:**

- Serão contabilizados 25 pontos referentes à entrega deste trabalho.
- Será avaliado individualmente a participação no envio de commits, o domínio sobre o assunto na entrevista.






**Elias Alves** @ Decom/FACET - UFVJM

[2] `http://pgdocptbr.sourceforge.net/pg80/client-authentication.html`
[3] `http://pgdocptbr.sourceforge.net/pg80/auth-methods.html`