INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SÃO PAULO - CAMPUS VOTUPORANGA

Bacharelado em Sistemas de Informação					
Disciplina:	Administração de Banco de Dados		Semestre:	5	Turma: A
Professor:	Dr. Evandro Jardini	Bimestre:1	Data:		
Aluno:			Num:		

Estudos de Casos para a Disciplina Administração de Banco de Dados

Grupo: No máximo 3 alunos(as).

Avaliação do trabalho: O trabalho será avaliado conforme orientação no Moodle. O **professor irá sentar-se junto** à maquina do grupo e acompanhará o grupo fazer:

1. Clonar na máquina local o **repositório no Github** criado exclusivamente para este trabalho.
2. Rodar no SGBD Postgres os *scripts* SQL clonados do repositório.

Avaliação de cada membro do grupo: O membro do grupo só terá nota caso haja pelo menos um arquivo postado no *Github* em seu nome. No trabalho, há 4 questões que exigem SQL. Cada membro deverá ser responsável por pelo menos um arquivo postado no *Github*. O membro do grupo será avaliado:

1. Caso não haja arquivo em seu nome, não haverá nota para o membro.
2. Caso algum *script* da questão 1 a 4 não execute, o aluno responsável por ele, terá nota igual a 0 (zero)
3. Após todos os *scripts* terem sido executados no Postgres, **o professor irá testar os exercícios da questão 1 a 4** por meio de comandos DML *select*, *insert*, etc para se certificar que o grupo criou os *scripts* de forma correta. Caso o professor encontre erro, o aluno responsável por resolver a questão irá perder a pontuação indicada.

Requisitos das regras do negócio: Um hospital deseja informatizar seus atendimentos para com os pacientes. Um dos requisitos é saber os dados dos pacientes e os médicos que lhes atenderam durante toda sua vida. Dos médicos também serão guardadas informações pertinentes a eles. Além disso, dos pacientes é necessário armazenar suas cirurgias, caso haja alguma.

O Diagrama Entidade Relacionamento (DER) para estes requisitos é o mostrado na figura 1. Na figura 2 têm-se os dados de alguns pacientes, médicos, cirurgias e atendimentos. Estes dados estão em fichas que deverão ser colocados dentro do banco de dados.

Requisitos técnicos para desenvolvimento do banco de dados: Na figura 3 é descrito o modelo relacional do DER. Note que há o uso de IDs e eles deverão ser **implementados com sequências** no SGBD.

- **Importante**: Notem que na figura 2 na tabela *Atende* tem-se os nomes dos pacientes e médicos e na tabela *Cirurgia* tem-se o nome dos pacientes, porém, na figura 3 não existe estes nomes, mas sim os IDs dos pacientes e médicos. Desta forma, considerando que os IDs foram implementados com *sequence* e o uso de *sequence* não permite controle de qual número será atribuído a um registro, quando vocês forem inserir os dados, não poderão inserir com ID fixo, mas sim, fazer um *select* na respectiva tabela pelo código do paciente ou médico e encontrar seu ID. Veja a seção sobre sequência na apostila para saber como fazer isso.

De acordo com os dados fornecidos, responda em SQL, as seguintes questões:

Figura 1: Diagrama Entidade Relacionamento da base de dados do hospital

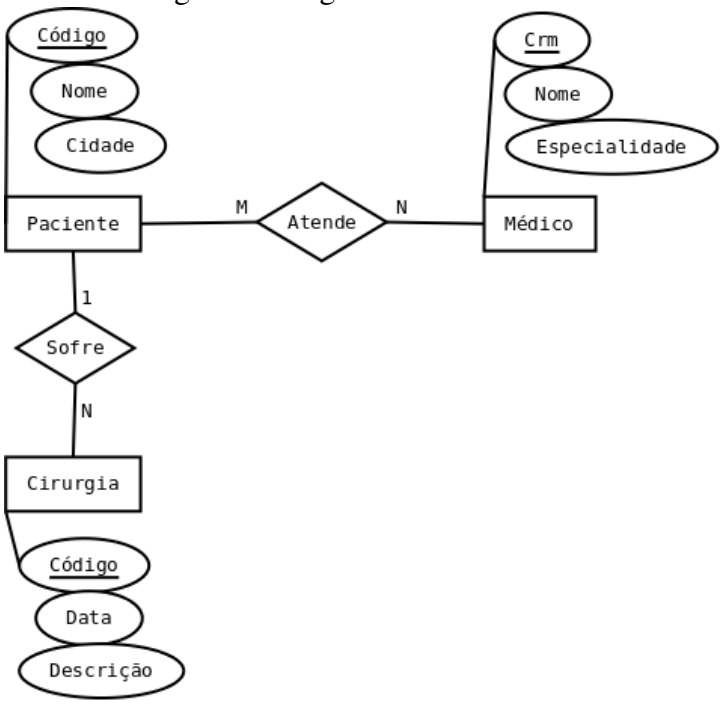

Figura 2: Dados do banco de dados do hospital

código	nome	idade
p1	João	12
p2	Maria	38
p3	Pedro	21
p4	Antônio	29

Paciente

crmv	nome	especialidade
m1	Marcos	Oftalmologista
m2	Tereza	Clínico Geral
m3	Paulo	Pediatra
m4	João	Clínico Geral

Médico

paciente	médico	data
João	Tereza	09/02/2008
Maria	Marcos	26/03/2006
Pedro	Paulo	11/09/2003
João	João	13/10/2007
Maria	Tereza	08/05/2008

Atende

código	data	descrição	Paciente
c1	25/07/2008	Apendicite	João
c2	07/02/2001	Retirada de cálculo renal	Maria
c3	14/11/2007	Unha encravada	Pedro
c4	23/01/2008	Implante de silicone	Maria

Cirurgia

Figura 3: Modelo relacional do banco de dados do hospital

paciente = {id_paciente, codigo, nome, idade}
medico = {id_medico, crm, nome, especialidade}
atende = {id_atende, id_paciente, id_medico, data}
cirurgia = {id_cirurgia, codigo, data, descrição, id_paciente}

1. (Se no momento do teste com comando DDL ocorrer erro, o aluno perde 80% da nota do trabalho) - Implemente o código SQL, em um único aquivo, para se criar (deverá ser criado um arquivo e postado no *Github*):

   (a) O usuário do sistema;

   (b) O banco de dados;

   (c) As sequencias para as tabelas;

   (d) As tabelas do sistema.

2. (Se no momento do teste com comando DML ocorrer erro, o aluno perde 80% da nota do trabalho) - Implemente uma função que informados o **código da cirurgia, código do paciente**, a **data da cirurgia** e a **descrição do procedimento cirúrgico**, crie uma nova cirurgia. É importante verificar a existência do paciente que irá fazer a cirurgia e caso o código do paciente exista, deve ser gerado um erro com RAISE e abortar a função (deverá ser criado um arquivo e postado no *Github*).

3. (Se no momento do teste com comando DML ocorrer erro, o aluno perde 80% da nota do trabalho) - O desempenho dos médicos é auferido pela quantidade de atendimento que fazem por período. Assim, desenvolva uma função que informadas as **datas inicial e final** dos atendimentos, retorne o nome do médico e a quantidade de atendimento que realizou no período. Retorne tudo isso em forma de registro (*returns setof record*). Exemplo: se você

colocou as datas 01/01/2003 a 31/12/2006, vai retornar Paulo 1; Marcos 1.
(deverá ser criado um arquivo e postado no *Github*)

4. (Se no momento do teste com comando DML ocorrer erro, o aluno perde 80% da nota do trabalho) - É importante realizar verificações de regras de negócio no momento em que dados são alterados ou inseridos. Desta forma, desenvolva um *trigger* que verifique, no momento em que uma cirurgia for inserida, se a data da cirurgia seja igual ou antes da data corrente. Se ela for maior, o *trigg*er não pode deixar que a cirurgia seja inserida. (deverá ser criado um arquivo e postado no *Github*)

5. (ENADE) Em um sistema gerenciador de banco de Postgres a coluna SALARIO da tabela COLABORADOR foi definida como NUMERIC(8,2) e a coluna JUROS foi definida como NUMERIC (2,2). Foi criada uma *view* com a seguinte expressão:
   CREATE VIEW LISTA_VALOR_JUROS
   AS SELECT NOME, SALARIO * JUROS as JUROS_MES
   FROM COLABORADOR
   Assinale a opção que apresenta o tipo de dado que suportará o resultado da coluna JUROS_MES criado na *view* resultante dessa expressão. A) NUMERIC (10,1) B) INTEGER C) NUMERIC (2,2) D) NUMERIC E) LONG INT

6. (ENADE) *Views* criadas nos bancos podem, de acordo com alguns critérios, **ser naturalmente atualizáveis**, o que significa, por exemplo, que podem ser objeto de comandos update do SQL sem a necessidade de mecanismos auxiliares ou *triggers*. Essa característica depende da expressão SQL que define a *view* e das tabelas/*views* de origem. Considere alguns tipos de construções SQL que podem ser empregadas na definição de uma coluna de uma view:
   I. funções de agregação, tais como sum, avg II. funções escalares, tais como sin, trim III. expressões aritméticas IV. expressões condicionais, tais como case V. subconsultas
   Está correto concluir que uma determinada coluna NÃO pode ser objeto de atualização quando resultar de qualquer dos tipos: A) apresentados, exceto I, II e III; B) apresentados, exceto III e IV; C) apresentados, exceto apenas V; D) apresentados, exceto V; E) apresentados.

7. (ENADE) Considere o seguinte trecho de código PLPGSQL com as linhas numeradas à esquerda.
   1 *$$*
   2 DECLARE
   3 nome1 VARCHAR (10);
   4 nome2 CHAR(9) ;
   5 BEGIN
   6 nome1 = 'IFSP';
   7 nome2 = 'IFSP';
   8 IF (nome1=nome2) THEN

9 RAISE NOTICE '% igual a %', nome1, nome2;

10 ELSE

11 RAISE NOTICE '% diferente a %', nome2, nome1;

12 END IF;

13END;

14$$

Ao executar este código no Postgres, em condições ideais,

A) será impresso a mensagem da linha 11. B) ocorrerá um erro, pois todos os strings deveriam estar delimitados por " " e não por ' '. C) ocorrerá um erro, pois nome1 e nome2 são de tipos diferentes. D) ocorrerá um erro, pois o comando de saída deveria ser RAISE EXCEPTION. E) será impresso a mensagem da linha 9 .