Prazo 30 de jun 23:59

Questão 1. Como todos sabem, o estado do Rio de Janeiro bem passando uma por uma severa crise financeira. Dessa forma, o governo estadual cogita diminuir horas de trabalho e salários de muitos funcionários. (fonte:<http://oglobo.globo.com/rio/rj-estuda-reduzir-jornada-salarios-de-servidores-do-estado-20358106>)

Você foi contratado para atualizar o banco de dados dos funcionários estaduais de acordo com as novas diretrizes. Considere a tabela FUNCIONARIO apresentada como base para os itens a seguir. As respostas podem ser dadas para os SGBDs mySQL, SQL Server, Oracle e PostgreSQL. Cada grupo deve enviar um zip ou rar com os scripts de criação dos objetos.

CREATE TABLE `funcionario` (

`nome` VARCHAR( 60 ) NOT NULL ,

`email` VARCHAR( 60 ) NOT NULL ,

`sexo` VARCHAR( 10 ) NOT NULL ,

`ddd` INT( 2 ) ,

'salario' NUMBER (2),

`telefone` VARCHAR ( 8 ) ,

'ativo' VARCHAR(1),

`endereco` VARCHAR( 70 ) NOT NULL ,

'cpf'VARCHAR (11) NOT NULL,

`cidade` VARCHAR( 20 ) NOT NULL ,

`estado` VARCHAR( 2 ) NOT NULL ,

`bairro` VARCHAR( 20 ) NOT NULL ,

`pais` VARCHAR( 20 ) NOT NULL ,

`login` VARCHAR( 12 ) NOT NULL ,

`senha` VARCHAR( 12 ) NOT NULL ,

`news` VARCHAR( 8 ) ,

`id` INT( 200 ) AUTO\_INCREMENT ,

UNIQUE (`id` ));

1. Elabore uma stored procedure para diminuir o salário de um funcionário em um determinado percentual. A sua procedure deve se chamar DiminuirSalario e deve receber como parâmetros de entrada o CPF do funcionário e um valor inteiro que representa o percentual de redução.

2. Além da redução de salários, o governo prevê demissões para funcionários que faltem sem justificativa apresentada. Esse tipo de controle não existe hoje no banco de dados. Sua tarefa é desenvolver um mecanismo que controle as faltas de cada um dos funcionários. A partir da 5a (quinta) falta sem justificativa o campo ATIVO da tabela funcionário deve ser setado para 'N" significando que ele foi demitido.

Sugestão: criem uma tabela que controle as faltas e justificativas e uma trigger associada a essa tabela para verificar a quantidade de faltas.

3. O governo do estado também deseja controlar todas as promoções dos funcionários ao longo do anos. Assim como no caso das faltas, esse mecanismo não se encontra implementado no banco de dados. É sua responsabilidade implementar esse controle. Cada funcionário possui um cargo (que por simplificação pode variar entre CARGO1, CARGO2 e CARGO 3) e seu nível pode variar entre 1 e 7. Ou seja, o funcionário pode ter o CARGO1 e Nível 5 no momento, e, na próxima promoção ele terá o CARGO1 (que não muda) e Nível 6, e assim por diante. Lembrando que cada funcionário só pode aumentar seu nível de 3 em 3 anos e não pode haver interseção de períodos entre dois níveis. Além disso, um funcionário só pode ser promovido para o nível imediatamente superior ao atual, logo uma promoção do Nível 1 para o Nível 3 é proibida. Desenvolva uma stored procedure que implemente a promoção de um determinado funcionário. Sua stored procedure deve receber o CPF do funcionário e o nível para promoção como parâmetros de entrada.

Questão 2. A seguir é apresentado o esquema relacional (script de criação de tabelas) relativo ao cadastramento de receitas e avaliações de receitas para o site da Palmirinha, famosa doceira da TV. Considere esse esquema para resolver os itens a seguir.

CREATE TABLE PESSOA

(

ID\_PESSOA integer NOT NULL,

NOME character varying,

LOGIN character varying,

SENHA character varying,

CONSTRAINT PK\_PESSOA PRIMARY KEY (ID\_PESSOA)

);

CREATE TABLE RECEITA

(

ID\_RECEITA integer NOT NULL,

DATA\_ENVIO date,

TITULO character varying,

MODO\_PREPARO character varying,

ID\_PESSOA integer,

CONSTRAINT PK\_RECEITA PRIMARY KEY (ID\_RECEITA),

CONSTRAINT FK\_PESSOA" FOREIGN KEY (ID\_PESSOA) REFERENCES PESSOA (ID\_PESSOA)

ON UPDATE NO ACTION ON DELETE NO ACTION

);

CREATE TABLE INGREDIENTE

(

ID\_INGREDIENTE integer NOT NULL,

DESCRICAO character varying,

NOME character varying,

MEDIDA character varying,

CONSTRAINT PK\_INGREDIENTE PRIMARY KEY (ID\_INGREDIENTE)

);

CREATE TABLE RECEITA\_INGREDIENTE

(

ID\_RECEITA integer NOT NULL,

ID\_INGREDIENTE integer NOT NULL,

QUANTIDADE double precision,

CONSTRAINT PK\_INGREDIENTE\_RECEITA PRIMARY KEY (ID\_RECEITA, ID\_INGREDIENTE),

CONSTRAINT FK\_INGREDIENTE FOREIGN KEY (ID\_INGREDIENTE)

REFERENCES INGREDIENTE (ID\_INGREDIENTE) ON UPDATE NO ACTION ON DELETE NO ACTION,

CONSTRAINT FK\_RECEITA FOREIGN KEY (ID\_RECEITA)

REFERENCES RECEITA (ID\_RECEITA) ON UPDATE NO ACTION ON DELETE NO ACTION

);

CREATE TABLE VOTACAO

(

ID\_VOTO integer NOT NULL,

NOTA double precision,

ID\_PESSOA integer,

ID\_RECEITA integer,

CONSTRAINT PK\_VOTO PRIMARY KEY (ID\_VOTO),

CONSTRAINT FK\_PESSOA FOREIGN KEY (ID\_PESSOA)

REFERENCES PESSOA (ID\_PESSOA) ON UPDATE NO ACTION ON DELETE NO ACTION,

CONSTRAINT FK\_RECEITA FOREIGN KEY (ID\_RECEITA)

REFERENCES RECEITA (ID\_RECEITA) ON UPDATE NO ACTION ON DELETE NO ACTION

);

1. Implemente uma stored procedure chamada INSERE\_RECEITA que insira uma nova receita no banco de dados da Palmirinha. Sua stored procedure deve receber como parâmetros o ID da pessoa que está postando a receita, o título da receita, a data de postagem e o modo de preparo e insira uma nova receita no banco de dados (tabela RECEITA). Além disso, a sua stored deve receber como parâmetros os nomes dos ingredientes utilizados na receita e o segundo as quantidades dos mesmos utilizadas na receita (assumindo a unidade padrão cadastrada na tabela INGREDIENTE campo MEDIDA).

Sugestão: use cursores ou tabelas temporárias.

Observação importante: deverá ser efetuado o teste para verificar se o ingrediente correspondente ao nome informado existe na tabela INGREDIENTE. Caso contrário, sua stored procedure deve exibir uma mensagem de erro. Atente também que na tabela RECEITA a referência ao ingrediente é realizada por meio do campo ID\_INGREDIENTE e não por seu nome.

2. Estenda a stored procedure INSERE\_RECEITA implementada no item 1 de forma que no momento da inserção de um ingrediente para uma receita, a mesma verifique a compatibilidade da quantidade do ingrediente informada na receita e sua medida. Por exemplo, ingredientes cuja medida seja "Unidade" só podem ter quantidades inteiras. Todas as outras medidas permitem valores fracionados. Caso a quantidade não seja compatível com a medida, sua stored procedure deve exibir uma mensagem informando que o valor foi aproximado por arredondamento para o inteiro mais próximo. Esse valor arredondado deve ser o armazenado no banco de dados.

3. Faça uma função que identifica entre as receitas cadastradas no último mês aquela que possui a maior média (calculada a partir do campo NOTA) dentre aquelas que tiveram pelo menos 1.000 votos. Em caso de empate a receita postada mais recentemente deve ser a selecionada.

4. Faça uma stored procedure/função que recebe um título de uma receita e verifica se a mesma pode ser fracionada por um fator (que é um parâmetro de entrada da função). Uma receita fracionada é aquela em que seus ingredientes podem ser divididos por aquele fator. Por exemplo, uma "meia-receita" se refere a metade da quantidade de cada ingrediente utilizado na receita completa. Para verificar se uma receita pode ser fracionada, devemos checar as medidas dos seus ingredientes. As medidas sempre possíveis de serem fracionadas são Kilograma, Grama e Litro. Ingredientes cuja medida seja "Unidade" só são fracionáveis por um fator caso sejam múltiplos desse fator. Por exemplo, se uma receita contiver 2 ovos e o fator informado for 0,5, esta receita poderá ser fracionada. Usando o fator 0,3 para a mesma quantidade de ovos, a resposta seria negativa. Sua stored procedure/função deve retornar o título da receita e o seu modo de preparo, além do nome e da quantidade fracionada de cada ingrediente.