Sistemas de Informação e Bases de Dados

Enunciado da Etapa 2 do Projeto de 2018/19

Ponto de partida

Considere como base de referência para esta segunda etapa do projeto o seguinte esquema relacional sobre a gestão de ginásios, especificado em SQL-DDL, e correspondente a uma simplificação dos conceitos definidos na primeira etapa:

```
-- ------------------------------------------------------------------------

DROP TABLE frequenta;
DROP TABLE aula;
DROP TABLE utente;

-- ------------------------------------------------------------------------

CREATE TABLE utente (
  nif        NUMBER(9),
  nome       VARCHAR2(80) CONSTRAINT nn_utente_nome   NOT NULL,
  genero     CHAR(1)      CONSTRAINT nn_utente_genero NOT NULL,
  ano        NUMBER(4)    CONSTRAINT nn_utente_ano    NOT NULL,
--
  CONSTRAINT pk_utente
    PRIMARY KEY (nif),
--
  CONSTRAINT ck_utente_nif
    CHECK (nif BETWEEN 1 AND 999999999),
--
  CONSTRAINT ck_utente_genero
    CHECK (genero IN ('F', 'M')),
--
  CONSTRAINT ck_utente_ano                -- Ano de nascimento do utente.
    CHECK (ano BETWEEN 1900 AND 2100)
);

-- ------------------------------------------------------------------------

CREATE TABLE aula (
  id          NUMBER(4),   -- Para referenciação mais simples.
  modalidade  VARCHAR2(40) CONSTRAINT nn_aula_modalidade  NOT NULL,
  espaco      VARCHAR2(40) CONSTRAINT nn_aula_espaco      NOT NULL,
  dia_semana  CHAR(3)      CONSTRAINT nn_aula_dia_semana  NOT NULL,
  hora_inicio NUMBER(2)    CONSTRAINT nn_aula_hora_inicio NOT NULL,
  hora_fim    NUMBER(2)    CONSTRAINT nn_aula_hora_fim    NOT NULL,
--
  CONSTRAINT pk_aula
    PRIMARY KEY (id),
--
  CONSTRAINT un_aula_espaco_tempo   -- Chave candidata.
    UNIQUE (espaco, dia_semana, hora_inicio, hora_fim),
--
  CONSTRAINT ck_aula_id
    CHECK (id BETWEEN 1 AND 9999),
--
  CONSTRAINT ck_aula_dia_semana  -- Pode ser preciso adaptar a outro idioma.
    CHECK (dia_semana IN ('SUN', 'MON', 'TUE', 'WED', 'THU', 'FRI', 'SAT')),
```

```
--
  CONSTRAINT ck_aula_hora_inicio
    CHECK (hora_inicio BETWEEN 0 AND 23),
--
  CONSTRAINT ck_aula_hora_fim
    CHECK (hora_fim BETWEEN 1 AND 24),
--
  CONSTRAINT ck_aula_inicio_fim
    CHECK (hora_inicio < hora_fim)
);

-- ------------------------------------------------------------------------

CREATE TABLE frequenta (
  utente,
  aula,
  data    DATE,
--
  CONSTRAINT pk_frequenta
    PRIMARY KEY (utente, aula, data),
--
  CONSTRAINT fk_frequenta_utente
    FOREIGN KEY (utente)
    REFERENCES utente (nif),
--
  CONSTRAINT fk_frequenta_aula
    FOREIGN KEY (aula)
    REFERENCES aula (id),
--
  CONSTRAINT ck_frequenta_data
    CHECK (TO_CHAR(data, 'YYYY') BETWEEN 1900 AND 2100)
    -- A conversão do ano para número é implícita.
);

-- ------------------------------------------------------------------------
```

As simplificações incluem: a) a remoção dos conceitos de ginásio, gerente, e treinador, passando a assumir-se que todos os utentes estão inscritos no mesmo ginásio, o qual está aberto durante todo o dia; b) o descarte das autorizações e adesões de utentes a modalidades, podendo um utente frequentar as aulas de qualquer modalidade; c) a eliminação do conceito de empresa, bem como das faturas, pagamentos, e preços das aulas; d) a possibilidade de um espaço poder ser usado para qualquer modalidade, com uma lotação máxima fixa, em vez de ser dependente da modalidade; e) a supressão de atributos de utente, espaço, e modalidade, tendo estes dois últimos ficado reduzidos ao nome; f) o deixar de ser preciso registar a hora de entrada de um utente numa aula, e se pôde ou não ter a aula; e g) as aulas começarem e terminarem a horas certas (ex. das dez às onze horas).

Objetivos

1. Codificar um pacote PL/SQL para a gestão do ginásio, cuja especificação de funcionalidades vem na secção seguinte.
2. Escrever um *script* com invocações das funções e procedimentos do pacote PL/SQL que demonstrem um cenário completo de gestão de dados do ginásio.
3. Traduzir para interrogações SQL os pedidos de dados indicados numa secção mais à frente.

Especificação de funcionalidades

As funções e procedimentos seguintes devem constar num pacote PL/SQL chamado `ginasio`.

`regista_utente(nif, nome, genero, ano_nascimento)`

Regista um utente com número de identificação fiscal, nome, género, e ano de nascimento.

`regista_aula(modalidade, espaco, dia_semana, hora_início, hora_fim) -> id`

Regista uma aula no mapa de aulas do ginásio, ou seja, semanalmente o espaço fica reservado para a modalidade, no dia de semana, entre as horas de início e fim. Caso não haja sobreposição da aula com outras existentes, é devolvido o seu identificador, para futura referenciação rápida.

`regista_frequencia_aula(utente, aula, data)`

Regista a ida de um utente a uma aula, numa data. A data é opcional, devendo ser assumida por omissão a data atual. O dia de semana da data tem de ser igual ao definido na aula, e existe um limite de 10 utentes por aula em cada data.

`lista_aulas_frequentadas(utente, data) -> cursor`

Devolve um cursor com as aulas frequentadas pelo utente durante o mês e ano da data (o dia é ignorado). Cada linha do cursor deve ter a data da aula, o dia de semana, as horas de início e de fim, o espaço do ginásio, e a modalidade.

`remove_utente(nif)`

Remove os dados de um utente e de todas as aulas que frequentou.

`remove_aula(aula)`

Remove uma aula do mapa de aulas, bem como todas as suas frequências por utentes.

`remove_frequencia_aula(utente, aula, data)`

Remove a ida de um utente a uma aula numa data.

Pedidos de dados

1. NIF e nome dos utentes que frequentaram aulas (pelo menos uma) em 2017, bem como as modalidades praticadas e as datas. O resultado não deve ter repetições e deve vir ordenado pela modalidade, nome, e NIF de forma ascendente, e pela data da aula de forma descendente.

2. Espaço e modalidade com aulas frequentadas por utentes que foram ao ginásio pelo menos uma vez ao domingo, ou então frequentadas por utentes do género masculino nascidos no século XXI, isto é, a partir de 2001, inclusive. Nota: pode usar construtores de conjuntos.

3. Dia de semana, horas de início e fim, espaço, e modalidade das aulas começadas ou terminadas da parte da tarde, e frequentadas em janeiro de 2018, por utentes com nome começado pela letra 'C', com idade entre os 17 e os 45 anos.

4. NIF, nome, e idade dos utentes que nunca frequentaram aulas de ginástica começadas de manhã. O resultado deve vir ordenado pela idade do utente de forma descendente.

5. Modalidade, espaço, dia de semana, e hora de início das aulas frequentadas por todas as utentes (género feminino) nascidas no ano em que nasceram mais utentes (ambos os géneros) do ginásio. No caso de existirem vários anos candidatos, deve ser escolhido o mais recente. O resultado deve vir ordenado pelo dia de semana, hora de início, e modalidade de forma ascendente.

6. Total pago e valor do IVA à taxa 23% das aulas frequentadas por cada utente, com NIF e nome, para cada ano e modalidade, assumindo que cada aula custa 10 euros. O resultado deve vir ordenado pelo NIF do utente e modalidade de forma ascendente, e pelo ano de forma descendente.
7. Para cada utente, com NIF e nome, calcular a soma das horas das aulas que já frequentou. Caso um utente não tenha ainda frequentado aulas, deve aparecer o valor zero na soma. O resultado deve vir ordenado de forma descendente pela soma das horas das aulas, e pelo NIF e nome do utente de forma ascendente. Dica: a função NVL pode ser útil.
8. NIF e nome dos utentes com maior quantidade de aulas frequentadas em cada ano, separadamente para homens e mulheres. Em caso de empate, devem ser mostrados todos os utentes em causa. O resultado deve vir ordenado pelo ano de forma descendente, e pelo género, NIF, e nome dos utentes de forma ascendente.

A ter em conta

- O pacote PL/SQL tem de compilar, o *script* de demonstração tem de executar na íntegra, e as interrogações SQL têm de ser aceites pelo SGBD; caso contrário, são *excluídos* da avaliação.
- Em caso de erro, as funções e procedimentos devem lançar *exceções* com texto explicativo.
- O pacote PL/SQL deve estar comentado, as variáveis e parâmetros devem ter nomes inteligíveis, e devem ser respeitadas as regras de alinhamento dos blocos de código.
- As interrogações SQL devem ser inteligíveis, seguindo a mesma apresentação dos exemplos das aulas teóricas, e sem terem mudanças bruscas de linha.
- As interrogações SQL devem ser as mais simples possíveis, evitando, por exemplo, a inclusão de tabelas desnecessárias na cláusula FROM ou o uso excessivo de sub-interrogações.
- Em caso de dúvida sobre a interpretação de valores duplicados no resultado, pode desambiguar acrescentando atributos à cláusula SELECT.

Estrutura e entrega do relatório

- O relatório em *formato digital* deve ser entregue através da atividade disponível na página de entrada da unidade curricular, num arquivo com nome SIBD-1819-TPXX-GYY-E2.ZIP, em que XX é o número da turma (ex. 11) e YY é o número do grupo (ex. 01).
- Dentro do arquivo devem estar: os ficheiros da especificação e código do pacote PL/SQL; o *script* que demonstra o uso do pacote num cenário de gestão de dados; e um ficheiro contendo os pedidos de dados em português e as correspondentes interrogações SQL.
- No início de todos os ficheiros deve constar a sigla da disciplina, a data, a etapa do projeto, o grupo, e o nome, número, e turma dos alunos.
- É também necessário *imprimir em papel* o ficheiro com os pedidos de dados e as interrogações SQL, e fazer a sua entrega na caixa de correio do professor das aulas teórico-práticas.
- As folhas devem ser agrafadas no canto superior esquerdo, não sendo precisa folha de rosto.
- O prazo de entrega do relatório é o final do dia **14 de dezembro de 2018**.

Bom trabalho na etapa 2 do projeto!