Exercícios de fixação

Deseja-se informatizar uma clínica médica, armazenando as informações de seus **pacientes**, dos **convênios** utilizados por estes **pacientes**, e dos exames realizados por eles.

- É necessário armazenar o nome, CPF, telefone e endereço de cada paciente;
- Os convênios possuem nome, CNPJ, nome e telefone da pessoa de contato;
- Os exames possuem um código e um **tipo**, além de um diagnóstico (texto) associado;
- Cada paciente pode realizar vários exames na clínica, sendo que cada exame pode ser feito por um convênio diferente;
- O sistema deve permitir a geração de um relatório mensal de todos os exames realizados no mês por um determinado convênio.

1) Defina formalmente o esquema relacional incluindo todos os atributos declarados no problema e também os atributos que serão necessários para satisfazer a todos os requisitos.

⇒ indique a chave primária e as restrições de integridade referencial de cada esquema de relação;

⇒ use a notação vista de conjunto vista em aula (apenas texto), ou use um editor online de diagramas relacionais (como o https://dbdiagram.io, http://draw.io, https://www.lucidchart.com, dentre outros)

Solução Exemplo

```
Paciente (CPF, nome, telefone, endereço)
Convenio(CNPJ, nome, telefone_contato)
PacienteConvenio(CPF, CNPJ)
Tipo_Exame(tipo, descricao)
Exame(código, CPF, CNPJ, tipo, diagnostico, data)

PacienteConvenio(CPF)[FK] → Paciente [PK]
PacienteConvenio(CNPJ)[FK] → Convenio[PK]
Exame(CPF,CNPJ)[FK] → PacienteConvenio[PK]
Exame(tipo)[FK] → Tipo_Exame[PK]
```

2) Para cada restrição de integridade que você definiu, exemplifique uma tupla inválida, isto é, que não satisfaça às regras estabelecidas em projeto;

Solução Exemplo

```
Paciente: Inserir uma tupla com CPF com valor null;
Convenio: Inserir uma tupla com CNPJ com valor null;
Tipo_Exame: Inserir uma tupla com tipo com valor null;
Exame: Inserir um tupla com CPF/CNPJ/Tipo com valores para qualquer
destes atributos não existindo nas relações referenciadas.
```

3) Escreva o correspondente código DDL para definição do esquema de dados;

Solução Exemplo

```
DROP TABLE Exame;
DROP TABLE PacienteConvenio;
DROP TABLE Paciente;
DROP TABLE Convenio;
DROP TABLE Tipo_exame;

CREATE TABLE Paciente(
        CPF             INTEGER NOT NULL,
        nome            VARCHAR2(256),
        telephone       VARCHAR2(256),
        endereco        VARCHAR2(256),

        CONSTRAINT PK_paciente PRIMARY KEY(CPF)
);

CREATE TABLE Convenio(
        CNPJ                            INTEGER NOT NULL,
        nome                            VARCHAR2(256),
        telefone_contato  VARCHAR2(256),

        CONSTRAINT PK_convenio PRIMARY KEY(CNPJ)
);

CREATE TABLE Tipo_exame(
       tipo             VARCHAR2(256) NOT NULL,
       descricao        VARCHAR2(256) NOT NULL,

   CONSTRAINT PK_tipo_exame PRIMARY KEY(tipo)
);

CREATE TABLE PacienteConvenio(
  CPF           INTEGER,
  CNPJ          INTEGER,
  CONSTRAINT PK_PacienteConvenio PRIMARY KEY(CPF, CNPJ)
);

CREATE TABLE Exame(
 codigo          INTEGER NOT NULL,
 diagnostico     VARCHAR2(256),
 data            DATE,
 CPF             INTEGER,
 CNPJ            INTEGER,
 Tipo            VARCHAR2(256),

 CONSTRAINT PK_exame PRIMARY KEY(codigo),
 CONSTRAINT UN_exame UNIQUE(CPF, CNPJ, Tipo),
 CONSTRAINT FK_paciente FOREIGN KEY(CPF, CNPJ) REFERENCES PacienteConvenio(CPF, CNPJ),
 CONSTRAINT FK_Tipo_Exame FOREIGN KEY(Tipo) REFERENCES Tipo_exame(tipo)
);
```

4) Escreva o código DML para inserir, pelo menos, duas tuplas em cada relação.

⇒ Execute o seu código SQL em sua de dados Oracle (ou PostgreSQL, ou MySQL).

Solução Exemplo

```
INSERT INTO paciente VALUES (456789788,'João', '16-99258-1111','Av.
Trabalhador São Carlense');
INSERT INTO paciente VALUES (123456795,'José', '16-99999-5522','Av. São
Carlos');

INSERT INTO convenio VALUES (225558018,'MED.São Carlos', '16-3322-5522');
INSERT INTO convenio VALUES (338700122,'Santa Casa', '16-3311-5522');
```

```
INSERT INTO tipo_exame VALUES (1,'Hemograma');
INSERT INTO tipo_exame VALUES (2,'Ressonância Magnética');

INSERT INTO PacienteConvenio VALUES(456789788, 225558018);
INSERT INTO PacienteConvenio VALUES(123456795, 338700122);

INSERT INTO exame(codigo, diagnostico, data, CPF, CNPJ, tipo) VALUES (1, 'Nada
a avaliar', TO_DATE('2021-07-02', 'yyyy/mm/dd'), 456789788, 225558018,1);
INSERT INTO exame(codigo, diagnostico, data, CPF, CNPJ, tipo) VALUES (2, 'Sem
prescrição aparente', TO_DATE('2021-07-02', 'yyyy/mm/dd'), 123456795,
338700122,2);
```

Considere os dados da Copa do Mundo (WorldCupMatches.csv e WorldCups.csv). Escreva o correspondente Esquema Relacional definindo o correspondente relacionamento entre os dois conjuntos de dados.

5) Defina formalmente o esquema relacional;

⇒ indique a chave primária e as restrições de integridade referencial de cada esquema de relação;

⇒ use a notação vista nas aulas (apenas texto) ou use um editor online de diagramas relacionais

(como o https://dbdiagram.io, http://draw.io, https://www.lucidchart.com, dentre outros)

Solução

```
WorldCupMatches(MatchID, Year, Datetime, Stage, Stadium, City,
HomeTeamName, HomeTeamGoals, AwayTeamGoals, AwayTeamName,
WinConditions, Attendance, HalfTimeHomeGoals, HalfTimeAwayGoals,
Referee, Assistant1, Assistant2, RoundID, HomeTeamInitials,
AwayTeamInitials)

WordlCup(Year, Country, Winner, RunnersUp, Third, Fourth, GoalsScored,
QualifiedTeams, MatchesPlayed, Attendance)

WorldCupMatches(Year)[FK] → WordlCup[PK]
```

6) Escreva o correspondente código DDL para definição do esquema de dados;

Solução Exemplo

```
DROP TABLE WorldCup CASCADE CONSTRAINTS;
CREATE TABLE WorldCup(
      year              INTEGER NOT NULL,
      country           VARCHAR2(50),
      Winner            VARCHAR2(50),
      RunnersUp         VARCHAR2(50),
      Third             VARCHAR2(50),
      Fourth            VARCHAR2(50),
      GoalsScored       INTEGER,
      QualifiedTeams    INTEGER,
      MatchesPlayed     INTEGER,
      Attendance        INTEGER,

      CONSTRAINT PK_worldcup PRIMARY KEY(year)
);

DROP TABLE WorldCupMatches CASCADE CONSTRAINTS;
CREATE TABLE WorldCupMatches(
      year                    INTEGER NOT NULL,
      datetime                DATE,
      Stage                   VARCHAR2(50),
```

```
      Stadium                VARCHAR2(50),
      City                   VARCHAR2(50),
      HomeTeamName           VARCHAR2(50),
      HomeTeamGoals          INTEGER,
      AwayTeamGoals          INTEGER,
      AwayTeamName           VARCHAR2(50),
      WinConditions          VARCHAR2(50),
      Attendance             INTEGER,
      HalfTimeHomeGoals      INTEGER,
      HalfTimeAwayGoals      INTEGER,
      Referee                VARCHAR2(50),
      Assistant1             VARCHAR2(50),
      Assistant2             VARCHAR2(50),
      RoundID                INTEGER,
      MachID                      INTEGER,
      HomeTeamInitials       VARCHAR2(5),
      AwayTeamInitials       VARCHAR2(5),

      CONSTRAINT PK_WCM PRIMARY KEY (matchID),
      CONSTRAINT FK_WCM_worldcup FOREIGN KEY(year) REFERENCES WorldCup(year)
);
```

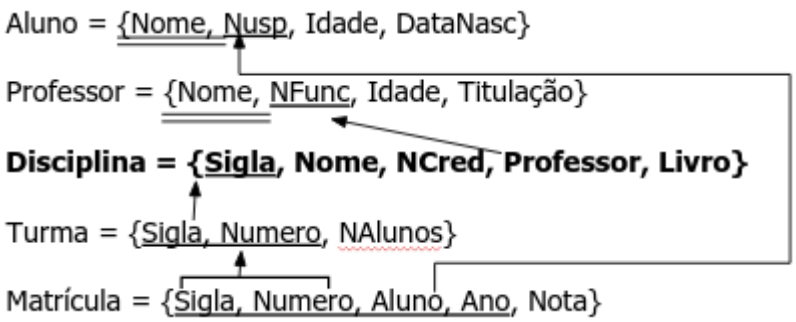
7) Escreva um programa python que lê os dados Copa do Mundo, escreve os correspondentes comandos INSERT; em seguida, execute os comandos gerados em sua base de dados Oracle (ou PostgreSQL, ou MySQL).

Solução Exemplo

```
import pandas as pd

##### atualizar os paths dos arquivos worldcupmathches.csv e worldcup.csv
worldcupmatches = pd.read_csv('WorldCupMatches.csv', encoding ='UTF-8' )
worldcup = pd.read_csv('WorldCups.csv', encoding ='UTF-8' )

insert_worldcup = []
insert_worldcupmatches = []
## table worldcup
for i in range(len(worldcup)):

    insert_worldcup.append("INSERT INTO WorldCup VALUES
({},'{}','{}','{}','{}','{}',{},{},{},{});".format(
          worldcup.loc[i,'Year'],
          worldcup.loc[i,'Country'],
          worldcup.loc[i,'Winner'],
          worldcup.loc[i,'RunnersUp'],
          worldcup.loc[i,'Third'],
          worldcup.loc[i,'Fourth'],
          worldcup.loc[i,'GoalsScored'],
          worldcup.loc[i,'QualifiedTeams'],
          worldcup.loc[i,'MatchesPlayed'],
          worldcup.loc[i,'Attendance'],
    ))


## table worldcupmatches
for i in range(len(worldcupmatches)):
    insert_worldcupmatches.append("INSERT INTO WorldCupMatches VALUES
({},TO_DATE('{}', 'dd mm yyyy -
hh24:mi'),'{}','{}','{}','{}',{},{},'{}','{}',{},{},{},'{}','{}','{}',{},{},'{
}','{}');".format(
         worldcupmatches.loc[i,'Year'],
         worldcupmatches.loc[i,'Datetime'],
         worldcupmatches.loc[i,'Stage'],
```

```
        worldcupmatches.loc[i,'Stadium'],
        worldcupmatches.loc[i,'City'],
        worldcupmatches.loc[i,'HomeTeamName'],
        worldcupmatches.loc[i,'HomeTeamGoals'],
        worldcupmatches.loc[i,'AwayTeamGoals'],
        worldcupmatches.loc[i,'AwayTeamName'],
        worldcupmatches.loc[i,'WinConditions'],
        worldcupmatches.loc[i,'Attendance'],
        worldcupmatches.loc[i,'HalfTimeHomeGoals'],
        worldcupmatches.loc[i,'HalfTimeAwayGoals'],
        worldcupmatches.loc[i,'Referee'],
        worldcupmatches.loc[i,'Assistant1'],
        worldcupmatches.loc[i,'Assistant2'],
        worldcupmatches.loc[i,'RoundID'],
        worldcupmatches.loc[i,'MatchID'],
        worldcupmatches.loc[i,'HomeTeamInitials'],
        worldcupmatches.loc[i,'AwayTeamInitials']
    ))
```

8) Usando Python ou o SQLDeveloper (ou o PGAdmin), carregue o esquema e os dados do Esquema Universidade.




```
1. Descompactar "EsquemaUniversidade.zip"
2. Rodar no SQLDeveloper "Universidade-Create.sql"
3. Rodar no SQLDeveloper "Universidade-Insert.sql"

-- CHECK create table e inserts (dados de apoio)
SELECT 'aluno', COUNT(*) FROM ALUNO UNION
SELECT 'PROFESSOR', COUNT(*) FROM PROFESSOR UNION
SELECT 'Disciplina', COUNT(*) FROM Disciplina UNION
SELECT 'Turma', COUNT(*) FROM Turma UNION
SELECT 'Matricula', COUNT(*) FROM Matricula;

/* Contagens Colhidas
'ALUNO'      COUNT(*)
---------- ----------
Disciplina         10
Matricula          46
PROFESSOR          5
Turma              20
aluno              10*/
```

Faça alterações no esquema

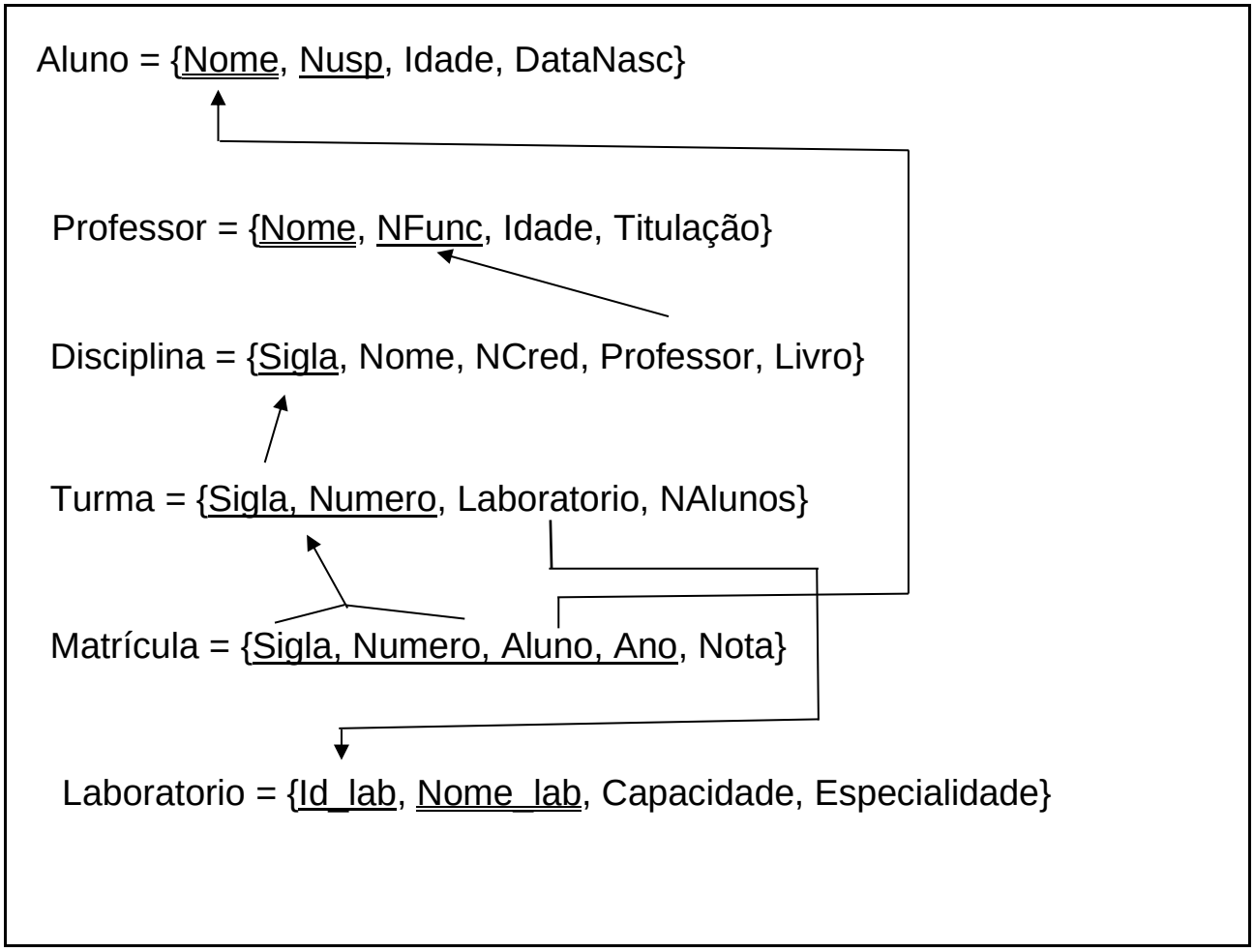
9) Adicione uma entidade Laboratório(id_lab, nome_lab, capacidade, especialidade);

```
CREATE TABLE LABORATORIO (
    ID_LAB NUMBER NOT NULL,
    NOME_LAB VARCHAR2(100) NOT NULL UNIQUE,
    CAPACIDADE NUMBER,
    ESPECIALIDADE VARCHAR2(255) NOT NULL,

    CONSTRAINT PK_Lab PRIMARY KEY (id_lab)
);
```

10) Altere a entidade Turma para que cada turma tenha um laboratório associado, sendo que um mesmo laboratório pode estar associado a mais de uma turma;

```
-- Adiciona o atributo laboratorio em TURMA
ALTER TABLE TURMA ADD laboratorio NUMBER;
-- Adiciona a restrição de chave estrangeira laboratorio em TURMA
ALTER TABLE TURMA ADD CONSTRAINT FK_LAB FOREIGN KEY (laboratorio) REFERENCES
LABORATORIO(ID_LAB) ON DELETE CASCADE;
```

11) Após as alterações, desenhe o diagrama relacional de todo o projeto.



12) Insira 4 laboratórios no banco de dados;

```
INSERT INTO LABORATORIO VALUES(1,'Laboratório de Inteligência
Computacional',55,'Inteligencia Artificial');
INSERT INTO LABORATORIO VALUES(2,'Laboratório de Aprendizado de
Rôbos',40,'Robotica');
INSERT INTO LABORATORIO VALUES(3,'Laboratório de Banco de Dados e
```

```
Imagens',40,'Banco de Dados e Processamento de Imagens');
INSERT INTO LABORATORIO VALUES(4,'Laboratório do Núcleo Interinstitucional de
Lingüística Computacional',55,'Inteligência Artificial (Linguística
Computacional)');
```

13) Associe as turmas existentes aos laboratórios que foram criados usando o comando UPDATE; deixe algumas turmas sem laboratório; e, pelo menos, um laboratório não vinculado a qualquer turma;

```
UPDATE TURMA SET laboratorio = 3 WHERE SIGLA = 'SC518' AND NUMERO = 1;
UPDATE TURMA SET laboratorio = 3 WHERE SIGLA = 'SC518' AND NUMERO = 2;
UPDATE TURMA SET laboratorio = 4 WHERE SIGLA = 'SA111' AND NUMERO = 1;
UPDATE TURMA SET laboratorio = 2 WHERE SIGLA = 'SS910' AND NUMERO = 1;
UPDATE TURMA SET laboratorio = 4 WHERE SIGLA = 'SS810' AND NUMERO = 2;
UPDATE TURMA SET laboratorio = 3 WHERE SIGLA = 'SE120' AND NUMERO = 2;
UPDATE TURMA SET laboratorio = 3 WHERE SIGLA = 'SM218' AND NUMERO = 1;

-- Laboratório id_lab = 1 não possui turmas
```

14) Formule uma consulta que relate quais turmas estão vinculadas a quais laboratórios, incluindo as turmas sem nenhum laboratório vinculado, e os laboratórios sem nenhuma turma vinculada;

```
-- FULL OUTERJOIN
SELECT T.SIGLA, L.id_lab, L.NOME_LAB
    FROM TURMA T FULL OUTER JOIN LABORATORIO L
    ON t.laboratorio = L.id_lab;
```

15) Formule uma consulta para computar o total de alunos vinculados a cada laboratório; por exemplo, se duas turmas estão vinculadas ao laboratório 1, uma com 50 alunos e outra com 40, o total de alunos vinculados ao laboratório é de 90 alunos;

Ordene o resultado pelo total de alunos.

⇒ Sua consulta terá uma junção, uma agregação, e uma ordenação.

```
SELECT L.nome_lab, SUM(T.NAlunos)
FROM LABORATORIO L JOIN TURMA T ON L.id_lab = T.laboratorio
GROUP BY nome_lab
ORDER BY SUM(T.NAlunos) DESC;
```

16) Formule uma consulta que retorne, para cada aluno, o nome da disciplina, o ano de todas as suas turmas e a nota obtida, o nome do respectivo professor, e o nome do laboratório vinculado. Desconsidere os alunos que não frequentam nenhum laboratório, isto é, basta um inner join.

```
SELECT A.NUSP, A.NOME, D.NOME, M.ANO, M.NOTA, P.NOME, L.nome_lab
FROM ALUNO A
    INNER JOIN MATRICULA M ON M.aluno = A.NUSP
    INNER JOIN TURMA T ON T.SIGLA = M.SIGLA and T.numero = M.NUMERO
    INNER JOIN LABORATORIO L ON T.laboratorio = L.id_lab
    INNER JOIN DISCIPLINA D ON T.SIGLA = D.SIGLA
    INNER JOIN PROFESSOR P ON P.NFUNC = D.PROFESSOR
    ORDER BY A.NOME;
```

```
- 4 Alunos não tiveram vínculos com nenhum laboratório.
```

Consulta aninhadas.
Liste a sigla e o nome das disciplinas que não possuem alunos matriculados:

17) Usando junção;

```
-- Left Outer Join
SELECT D.Sigla, D.Nome
FROM Disciplina D LEFT OUTER JOIN MATRICULA M ON D.sigla = M.sigla
WHERE M.aluno is null;
```

18) Usando consulta aninhada correlacionada (EXISTS).

```
-- NOT EXISTS
SELECT D.Sigla, D.Nome
FROM Disciplina D
WHERE NOT EXISTS
 (SELECT NULL
  FROM MATRICULA M
  WHERE D.sigla = M.sigla)
```

19) Usando consulta aninhada não-correlacionada (IN)

```
SELECT D.Sigla, D.Nome

FROM Disciplina D

where D.Sigla NOT IN

                  (select M.Sigla

                  from Matricula M)
```

Em uma conexão MongoDB

20) Crie uma coleção denominada Universidade;

```
db.createCollection("Universidade");
```

21) Crie um documento que Inclua todas as informações do esquema Universidade, isto é:

```
Aluno{
      Nome
      NUSP
      Idade
      DataNasc
      Matricula{
            Disciplina{
                  Sigla
                  Nome
                  NCred
                  Livro
```

```
                        Professor{
                              NFunc
                              Nome
                              Idade
                              Titulacao
                        }
                  }
                  Numero_da_turma
                  Num_alunos_da_turma
                  Ano
                  Nota
            }
      }
```

```
db.Universidade.insert({
Nome: "Aparecido Carvalho",
NUSP: 32001,
Idade: 30,
DataNac: "1985-08-26",
Matricula:{
     Disciplina:{
            Sigla:"SCC001",
            Nome:"Introução a Programação",
            NCred : 5,
            Livro: "Python - Introduction",
            Professor:{
                  NFunc: 601234,
                  Nome : "José da Silva",
                  Idade : 40,
                  Titulação :"Doutorado em Ciência da Computação"}
      },
      Numero_da_turma: 1,
      Num_alunos_da_turma: 100,
      Ano: 2021,
      Nota:7
    }
  }
)

-- visualizar a inserção
db.Universidade.find()
```

22) Observando a estrutura do documento Aluno, identifique duas situações nas quais uma situação de inconsistência pode ocorrer com relação a outros documentos que possam ser inseridos na base de dados.

```
   1. Disciplinas com mesma sigla e nomes diferentes;
   2. Mesma disciplina cadastrada com professores diferentes;
   3. Um mesmo aluno matriculado em outra disciplina via outro documento, mas
      com dados (NUSP, Nome, Idade, etc) diferentes;
   4. Uma mesma disciplina aparecendo em documentos de diferentes alunos, mas
      com dados (NCred, Livro, etc) diferentes;
   5. Um mesmo professor aparecendo em documentos de diferentes alunos, mas
      com dados (Nome, Idade, Titulação) diferentes.
Entre outras possibilidades.
```

23) Em MongoDB, e em outras soluções NoSQL, a escalabilidade horizontal é conseguida por meio de:
a) Múltiplos discos rígidos organizados por meio de tecnologia RAID;

b) Pela instalação de mais pentes de memória, espaço em disco, processadores com múltiplos cores, e largura de banda ampliada;

c) Pela técnica denominada Sharding, a qual prevê o particionamento horizontal (subconjuntos de documentos) dos dados, com cada partição sendo associada a um nó de armazenamento e processamento;

d) Contratando-se mais administradores de bancos de dados, os quais deverão ser alocados em uma mesma unidade física das instalações da empresa.

JUSTIFICATIVA: as alternativas a) e b) referem-se a técnicas de escalabilidade vertical; a alternativa d) não faz sentido no contexto da questão.