EACH – Escola de Artes, Ciências e Humanidades
Curso de Sistemas de Informação
ACH 2004 Banco de Dados
Prof. José de J. Pérez-Alcáza

PRIMEIRA PROVA DE BANCO DE DADOS - **Turma 94/2020**

Nome: Arthur Font Gouveia N.USP: 12036152

1. O Clube Globetrotters é organizado em capítulos. O presidente de um capítulo nunca pode servir como o presidente de qualquer outro capítulo, e cada capítulo dá ao seu presidente algum salário. Capítulos continuam se movendo para novos locais e um novo presidente é eleito quando (e somente quando) um capítulo se move. Esses dados são armazenados em uma relação G(C, S, L, P), onde os atributos são capítulos (C), salários (S), localizações (L) e presidentes (P). A consulta a seguir é feita com freqüência e você deve ser capaz de respondê-la eficientemente: "Quem era o presidente do capítulo X quando estava no local Y? . Dadas a seguintes dependências funcionais: PC → S; CL → P; e P → C.

   a) Defina quais são as chaves candidatas para a relação G?

   R: As chaves candidatas para a relação G são CL e PL. Pois a partir de CL é possível determinar todos os atributos e PL → CL por aumento da dependência funcional P → C.

   b) Em qual forma normal está a relação G? Crie um esquema de banco de dados normalizado para o clube. Justifique às suas decisões. 20%

   R: A relação G está na 1 FN, pois todos os atributos são simples. A relação G não está na 2 FN pois C é dependente funcional parcial de PL.

   Um esquema de banco de dados normalizado para o clube está representado a seguir. Este esquema evita redundância já que o presidente é determinado pela chave (Capítulo, Local) e visto que um capítulo dá ao seu presidente um salário, é possível obter o salário através do presidente.

   G1, onde CL é a chave primária

C	L	P

   G2, onde P é a chave primária

P	S

2. Considere a relação EXAME abaixo que possui o seguinte significado: Um estudante é examinado em uma disciplina e obtém uma posição na lista de classe. Sabe-se que dois estudantes não podem obter a mesma posição em uma mesma disciplina. A relação EXAME está em 3 FN? E em BCNF? Justifique sua resposta. 20%

   R: A relação EXAME está em 3 FN pois não há dependencia funcional transitiva. A relação EXAME também está em BCFN pois não há atributos que dependam de atributos não chave.

EXAME

Estudante	Disciplina	Posição
100	Matemática	7
100	Física	7
200	Matemática	3
200	Física	8

3. As seguintes relações mantem informações sobre voos de companhias aéreas. 30%

Estudante(estid: integer, estnome: string, areaPesq: string, nível: string, idade: integer)
Aula(anome: string, dia_hora: string, sala: string, profid: integer)
Matriculado(estid: integer, anome: string)
Professor(profid: integer, profnome: string, deptoid: integer)

O significado das relações é claro; por exemplo. Matriculado tem uma tupla por cada par estudante-aula tal que o estudante está matriculado na aula. O atributo dia_hora tem um formato definido da seguinte forma: “Sexta; 14:00” (“Nome do dia; hora em formato 24 hs.). Escreva as seguintes consultas em SQL. Nenhuma duplicata deveria ser impresa em qualquer uma das respostas.

a. Achar a idade do estudante mais velho que tem como área de pesquisa (areaPesq) História ou matriculado em uma aula ensinada por “Albert Einstein”.

R:

```
SELECT MAX(idade)
FROM Estudante
LEFT OUTER JOIN Matriculado ON Matriculado.estid = Estudante.estid
LEFT OUTER JOIN Aula ON Aula.anome = Matriculado.anome
LEFT OUTER JOIN Professor ON Aula.profid = Professor.profid
WHERE areaPesq = "Historia" OR Professor.profnome = "Albert Einstein";
```

b. Achar os nomes de todas as aulas que acontecem na sala R128 ou têm 5 ou mais estudantes matriculados.

R:

```
SELECT Aula.anome
FROM Aula
LEFT OUTER JOIN Matriculado ON Aula.anome = Matriculado.anome
GROUP BY Aula.anome
HAVING COUNT(Aula.anome) >= 5 OR sala = "R128"
```

c. Achar os nomes dos estudantes que estão matriculados em duas aulas que acontecem ao mesmo tempo.

R:

```
SELECT estnome
FROM Estudante JOIN Matriculado ON Estudante.estid = Matriculado.estid
JOIN Aula ON Matriculado.anome = Aula.anome
GROUP BY estid, dia_hora
HAVING COUNT(estid) >= 2
```

d. Achar o nome dos professores que ensinam aulas com menos de 5 matriculados.

R:

```
SELECT DISTINCT (profnome)
FROM Professor JOIN Aula ON Professor.profid = Aula.profid
JOIN Matriculado ON Matriculado.anome = Aula.anome
GROUP BY Professor.profid, Aula.anome
HAVING COUNT(profnome) < 5
```

e. Achar o nome dos estudantes matriculados no máximo número de aulas.

R:

```
SELECT estnome
FROM Estudante JOIN Matriculado ON  Estudante.estid = Matriculado.estid
GROUP BY estid
ORDER BY COUNT(estid) DESC LIMIT 1
```

Observação: Considerei que o enunciado está pedindo os estudantes que estão matriculados no maior número de aulas. Porém também é possível interpretar “os estudantes que estão matriculados em todas as aulas do banco de dados”.

4. Considere o seguinte esquema relacional. Um empregado pode trabalhar em mais de um departamento: o campo pct_tempo da relação Trabalha apresenta o percentual de tempo que um empregado trabalha em um departamento dado. 30%

   Emp(empid: integer, enome: string; idade: integer, salario: real)
   Trabalha(empid: integer, deptid: integer, pct_tempo: integer)
   Dept(deptid: integer, orcamento: real, gerenteid: integer)

   Escreva restrições de integridade (de domínio, chave, chave estrangeira, ou CHECK, ou assertions) ou triggers (em PostgreSQL ou Oracle) ou rules em PostgreSQL, para reforçar cada um dos seguintes requisitos considerados independentemente.

   a. Defina uma restrição de tabela que assegure que os empregados devem ter um salário máximo de R$20000.

```
CREATE TABLE Emp (
    …
        CHECK (salario <= 20000)
    );
```

   b. Defina uma restrição de tabela sobre Dept que assegure que todo gerente deve ser também um empregado.

```
CREATE ASSERTION gerente_empregado
CHECK (NOT EXISTS (SELECT gerenteid
                   FROM Dept
                   LEFT OUTER JOIN Emp ON Dept.gerenteid = Emp.empid
                   WHERE gerenteid != empid));
```

   c. O percentual total de trabalho de um empregado deve ser abaixo do 100%

```
CREATE ASSERTION percentual_total
CHECK (NOT EXISTS (SELECT empid
                   FROM Trabalha
                   GROUP BY empid
                   HAVING SUM (pct_tempo) > 100));
```

   d. Um gerente deve sempre ter um salário maior que o de qualquer empregado que ele ou ela gerencia

```
CREATE ASSERTION salario_gerente
CHECK (NOT EXISTS (SELECT empid
                   FROM Emp
                   WHERE Emp.salario >  (SELECT Emp.salario
                                         FROM Emp JOIN Dept ON
                                           Emp.empid = Dept.gerenteid
                                         JOIN Trabalha ON
                                           Dept.deptid = Trabalha.deptid)));
```

   e. Toda vez que a um empregado recebe um aumento, o salário do seu gerente deve ser incrementado para ser no mínimo igual. Além disso, toda vez que a um empregado recebe um aumento, o orçamento do departamento deve ser incrementado para ser maior que a soma dos salários de todos os empregados do departamento.

```
CREATE RULE  salario_update AS
ON UPDATE TO Emp WHERE NEW.salario > OLD.salario
                   AND NEW.salario > (SELECT Emp.salario
                                      FROM Emp JOIN Dept ON
                                        Emp.empid = Dept.gerenteid
                                      JOIN Trabalha ON
```

```
                                                    Dept.deptid = Trabalha.deptid)));

 DO ALSO UPDATE Emp SET salario =  NEW.salario
                                WHERE empid = (SELECT Emp.empid
                                                FROM Emp JOIN Dept ON
                                                Emp.empid = Dept.gerenteid);
DO ALSO UPDATE Dept SET orcamento =  orcamento + NEW.salario;
```