Trabalho de SQL

Disciplina: INE5423 – Banco de Dados I
Professor: Ronaldo S. Mello

Alunos:
Anselmo Soethe Nurnberg Junior 07232128
Ramon Dutra Miranda - 07232120

- Criação das Tabelas

```
CREATE TABLE Filmes (
ID        INT NOT NULL UNIQUE CHECK (ID > 0),
titulo    VARCHAR(40)NOT NULL,
```

```
duracao          NUMERIC(3) NOT NULL CHECK (duracao > 0),
ano      NUMERIC(4) NOT NULL CHECK (ano > 1895),
categoria     INT NOT NULL,
estilo    INT NOT NULL,
PRIMARY KEY(ID),
FOREIGN KEY(categoria) REFERENCES Categorias,
FOREIGN KEY(estilo) REFERENCES Estilos
)

CREATE TABLE Categorias (
ID       NUMERIC(1) NOT NULL UNIQUE CHECK (ID > 0),
nome             VARCHAR(10)NOT NULL CHECK (nome IN 'lançamento', 'normal', 'acervo'),
preco            NUMERIC(4,2) NOT NULL CHECK (preco > 0),
PRIMARY KEY(ID)
)

CREATE TABLE Estilos (
ID       NUMERIC(2) NOT NULL UNIQUE CHECK (ID > 0),
nome             VARCHAR(10)NOT NULL CHECK (nome
IN 'ação', 'drama', 'comédia', 'aventura', 'musical', 'ficção', 'infantil',
'documentário', 'suspense', 'romance' , 'terror'),
PRIMARY KEY(ID)
)

CREATE TABLE Copias (
ID       INT NOT NULL CHECK (ID > 0),
filme    INT NOT NULL,
midia    VARCHAR(3) NOT NULL CHECK (midia IN 'DVD','VHS'),
tipo     VARCHAR(9) NOT NULL CHECK (tipo IN 'dublado','legendado'),
PRIMARY KEY(ID, filme),
FOREIGN KEY(filme) REFERENCES Filmes
)

CREATE TABLE Clientes (
ID       INT NOT NULL UNIQUE CHECK (ID > 0),
nome             VARCHAR(40)NOT NULL,
endereco   VARCHAR(140),
fone     NUMERIC(10) CHECK (fone > 0),
cidade       VARCHAR(40),
responsavel    INT NOT NULL,
CPF      NUMERIC(11) NOT NULL UNIQUE,
PRIMARY KEY(ID),
FOREIGN KEY(responsavel) REFERENCES Clientes
)
```

```
CREATE TABLE Locacoes (
ID        INT NOT NULL UNIQUE CHECK (ID > 0),
filme     INT NOT NULL,
cliente     INT NOT NULL,
dataR             DATE NOT NULL ,
dataPD         DATE NOT NULL CHECK(dataR <= dataPD),
funcionario    INT NOT NULL,
dataD             DATE CHECK (dataD >= dataR),
PRIMARY KEY(ID, filme, dataR),
FOREIGN KEY(ID) REFERENCES Copias,
FOREIGN KEY(filme) REFERENCES Copias,
FOREIGN KEY(cliente) REFERENCES Clientes,
FOREIGN KEY(funcionario) REFERENCES Funcionarios
)

CREATE TABLE Reservas (
ID        INT NOT NULL UNIQUE CHECK (ID > 0),
filme     INT NOT NULL,
cliente     INT NOT NULL,
dataR             DATE NOT NULL ,
dataPD         DATE NOT NULL CHECK (dataR <= dataPD),
funcionario    INT NOT NULL,
dataRes           DATE NOT NULL CHECK (dataRes <= DataR),
PRIMARY KEY(ID, filme, dataR),
FOREIGN KEY(ID) REFERENCES Copias,
FOREIGN KEY( filme) REFERENCES Copias,
FOREIGN KEY(cliente) REFERENCES Clientes,
FOREIGN KEY(funcionario) REFERENCES Funcionarios
)

CREATE TRIGGER DeletaHistorico AFTER UPDATE ON Reservas
(DELETE FROM Reservas WHERE dataR > day(getdate()))

CREATE TABLE Funcionarios (
ID        INT NOT NULL UNIQUE CHECK (ID > 0),
nome              VARCHAR(40)NOT NULL,
endereco    VARCHAR(140),
fone      NUMERIC(10) CHECK(fone > 0),
cidade         VARCHAR(40),
salario     NUMERIC(5,2) NOT NULL CHECK(salario > 0),
turno     VARCHAR(1) NOT NULL CHECK (turno IN 'M', 'T', 'N'),
CPF      NUMERIC(11) NOT NULL UNIQUE,
PRIMARY KEY(ID)
```

)

- Consultas

/*1) buscar os IDs e títulos dos filmes dos anos 2008 e 2009 com duração superior a 90 e inferior a 120. Exibir o resultado ordenado por título;*/

SELECT ID, titulo FROM Filmes WHERE (ano BETWEEN 2008 AND 2009) AND (duracao BETWEEN 90 AND 120) ORDER BY titulo

/*2) buscar os IDs e nomes dos funcionários do noturno que também são clientes e realizaram reservas em 01/11/2010;*/

SELECT f.ID, f.nome FROM Funcionarios AS f JOIN Clientes AS c ON f.CPF = c.CPF JOIN Reservas as r ON c.ID = r.cliente WHERE f.turno = 'N' AND r.dataRes = '2010-11-01'

/*3) buscar os títulos e nomes de estilos dos filmes locados em 30/09/2010 e 01/11/2010;.Exibir o resultado ordenado de forma decrescente por estilo e de forma crescente por título;*/???

SELECT f.titulo, e.nome FROM Filmes AS f JOIN Estilos AS e ON f.estilo = e.ID JOIN Locacoes AS l ON f.ID = l.filme WHERE (l.dataR = '2010-09-30' OR l.dataR = '2010-11-01') ORDER BY e.nome DESC, f.titulo ASC

/*4) buscar os IDs e títulos dos lançamentos, e para aqueles que foram locados, exibir o número de vezes que foram locados;*/

SELECT f.ID, f.titulo, count(*) AS Nlocados FROM Filmes AS f JOIN Categorias AS c ON f.categoria = c.ID LEFT JOIN locacoes AS l ON f.ID = l.filme WHERE c.nome='lancamentos' GROUP BY f.ID

/*5) buscar os nomes e endereços dos clientes de Florianópolis e os nomes dos clientes que eles são responsáveis;*/

SELECT c1.nome , c1.endereco, c2.nome FROM Clientes AS c1 JOIN Clientes AS c2 ON c1.ID = c2.responsavel WHERE c1.cidade = 'Florianópolis'

/*6) buscar os nomes e endereços dos clientes que já entregaram DVDs com atraso;*/

SELECT c.nome, c.endereco FROM Locacoes AS l NATURAL JOIN Copias AS cop JOIN Clientes AS c ON l.cliente = c.ID WHERE cop.media = 'DVD' AND l.dataD > l.dataPD

/*7) buscar os nomes, cidades e endereços dos funcionários do diurno (manhã e tarde) e dos

clientes com reserva;*/

SELECT nome, cidade, endereco FROM Funcionarios WHERE turno = 'M' OR turno = 'T'
UNION
SELECT c.nome, c.cidade, c.endereco FROM Clientes AS c JOIN Reservas AS r ON c.ID = r.cliente

/*8) buscar as identificações (ID+filme) das cópias do filme X-Men 3 que estão disponíveis para locação ou reserva em 15/11/2010;*/

SELECT c.ID, c.filme FROM Copias AS c JOIN Filmes AS f ON f.ID = c.filme WHERE f.titulo = 'X-Man 3' AND c.ID NOT IN (SELECT r.ID FROM Reservas AS r UNION SELECT l.ID FROM Locacoes AS l WHERE dataR = '2010-11-20')

/*9) buscar os IDs, nomes e fones dos clientes que já locaram tanto filmes em VHS quanto filmes em DVD;*/

SELECT c.ID, c.nome, c.fone FROM Locacoes AS l NATURAL JOIN Copias AS cop JOIN Clientes AS c ON c.ID = l.cliente WHERE COUNT(c.tipo = 'DVD') = COUNT(c.tipo = 'VHS') GROUP BY c.ID

/*10) buscar os IDs e títulos dos lançamentos que possuem uma única cópia;*/

SELECT f.ID, f.titulo FROM Filmes AS f JOIN Categorias AS c ON f.categoria = c.ID JOIN Copias AS cop ON f.ID = cop.filme WHERE c.nome = 'lançamento' AND COUNT(cop.ID) = 1 GROUP BY f.ID

/*11) buscar pares de identificadores de cópias diferentes que pertencem ao mesmo filme, sem repetir um mesmo par na resposta;*/

SELECT c1.ID , c1.filme, c2.ID, c2.filme FROM Copias AS c1 JOIN Copias AS c2 ON c1.filme = c2.filme WHERE c1.ID != c2.ID

/*12) buscar o ID e o nome do funcionário que realizou o maior número de locações;*/

SELECT f.ID, f.nome FROM Funcionarios AS f JOIN Clientes AS c ON f.CPF = c.CPF JOIN Locacoes AS l ON c.ID = l.cliente GROUP BY COUNT(c.ID) DESC LIMIT 1

/*13) buscar os IDs, nomes e fones dos clientes que locaram apenas filmes de ação e de suspense;*/

SELECT c.ID, c.nome , c.fone FROM Clientes AS c WHERE c.ID NOT IN (SELECT l.cliente FROM Locacoes AS l JOIN Filmes AS f ON l.filme = f.ID JOIN Estilos AS e ON e.ID = f.estilo WHERE e.nome != 'ação' AND e.nome != 'suspense')

/*14) buscar os IDs, nomes e fones dos clientes de Florianópolis que já locaram todos os VHS legendados;*/

SELECT c.ID, c.nome, c.fone FROM Locacoes AS l NATURAL JOIN Copias AS cop JOIN Clientes AS c ON c.ID = l.cliente WHERE c.cidade = 'Florianópolis' AND cop.midia = ALL(SELECT media FROM Copias WHERE media = 'VHS' AND tipo ='legendado')

/*15) buscar os IDs e títulos dos filmes de ação que possuem duração superior a duração de todos os filmes de suspense, terror e drama;*/

SELECT f.ID, f.titulo FROM Filmes AS f JOIN Estilos AS e ON f.estilo = e.ID WHERE e.nome = 'ação' AND MAX(f.duracao) > ALL(SELECT f2.duracao FROM Filmes AS f2 JOIN Estilos AS e2 ON f2.estilo = e2.ID WHERE e2.nome = 'suspense' OR e2.nome = 'terror' OR e2.nome = 'drama')

/*16) remover os clientes cuja última retirada de filme para locação ocorreu em data igual ou anterior a 31/12/2000;*/

DELETE FROM Clientes WHERE ID NOT IN (SELECT l.cliente FROM Locacoes AS l WHERE l.dataR > '2000-12-31')

/*17) remover as cópias VHS dubladas que nunca foram reservadas;*/

DELETE FROM Copias AS c WHERE c.media = 'VHS' AND c.tipo = 'dublado' AND c.ID, c.filme NOT IN (SELECT r.ID, r.filme FROM Reservas AS r) /*essa consulta é pra sempre voltar vazia, pois pelas especificações da tabela Reserva ela não guarda Histórico*/

/*18) os funcionários que não locaram ou reservaram filmes no mês de outubro de 2010 tiveram uma redução de 20% nos seus salários.*/

UPDATE Funcionarios SET salario = salario-salario*0.2 WHERE CPF NOT IN (SELECT c.CPF FROM Clientes AS c JOIN Locacoes AS l ON l.cliente = c.iD WHERE EXTRACT(MONTH FROM TIMESTAMP l.dataR) = 10 AND EXTRACT(YEAR FROM TIMESTAMP l.dataR) = 2010 UNION SELECT c2.CPF FROM Clientes AS c2 JOIN Reservas AS r ON r.cliente = c2.iD WHERE EXTRACT(MONTH FROM TIMESTAMP r.dataR) = 10 AND EXTRACT(YEAR FROM TIMESTAMP r.dataR) = 2010)