Exercício 1:
Selecione 15 consultas das listas de exercícios anteriores e crie uma visão para cada uma delas.

a) CREATE OR REPLACE RULE compra_produto AS
ON INSERT TO cp DO ALSO
SELECT DISTINCT p.den_produto
FROM produto AS p,
compra AS c,
compra_produto AS cp
WHERE p.id_produto = cp.id_produto
AND c.id_compra = cp.id_compra
AND EXTRACT(year FROM c.data) = 2010
ORDER BY p.den_produto;

b) CREATE OR REPLACE RULE valor_total AS
ON INSERT TO cp DO ALSO
SELECT EXTRACT(year FROM data) || '-' || DATE_PART('month', data) AS mes,
SUM(valor_total) AS total
FROM venda
GROUP BY EXTRACT(year FROM data) || '-' || DATE_PART('month', data)
ORDER BY mes;

c) SELECT c.den_cliente,
SUM(cr.valor) AS total
FROM cliente AS c,
conta_receber AS cr
WHERE c.id_cliente = cr.id_cliente
AND pago = 'N'
GROUP BY c.den_cliente;

d) SELECT f.den_fornecedor,
COUNT(*) AS contas_pagar
FROM fornecedor AS f,
conta_pagar AS cp
WHERE f.id_fornecedor = cp.id_fornecedor
GROUP BY f.den_fornecedor
HAVING COUNT(*) > 10;

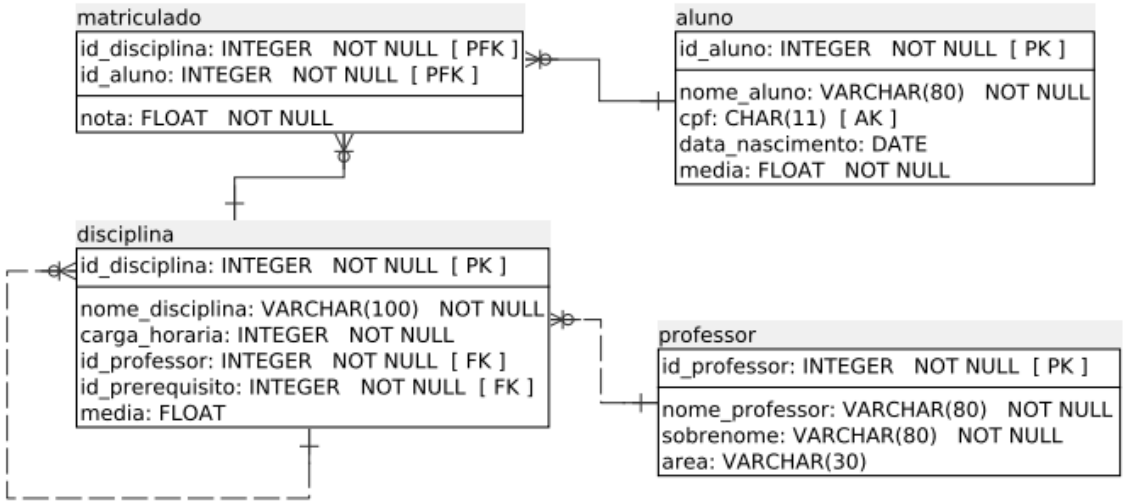
e) SELECT *
FROM conta_pagar
WHERE pago = 'N'
AND data_vencimento <= '2011-12-31';

f) SELECT c.den_cidade AS cidade,
SUM(v.valor_total) AS total
FROM cidade AS c,
cliente AS cl,
venda AS v
WHERE c.id_cidade = cl.id_cidade

```
AND cl.id_cliente = v.id_cliente
GROUP BY c.den_cidade;

g) SELECT f.den_fornecedor,
SUM(c.valor_total) AS total
FROM fornecedor AS f,
compra AS c
WHERE f.id_fornecedor = c.id_fornecedor
GROUP BY f.den_fornecedor;

h) SELECT id_disciplina, nome_disciplina
FROM disciplina AS d
WHERE NOT EXISTS (
SELECT id_displina
FROM matriculado AS m
WHERE m.id_disciplina = d.id_disciplina);

i) SELECT a1.nome_aluno, a2.nome_aluno
FROM aluno AS a1,
aluno AS a2
WHERE a1.id_aluno <> a2.id_aluno;

j) SELECT id_professor, nome_professor
FROM professor AS p
WHERE NO EXISTS (
SELECT d.id_professor
FROM disciplina AS d
WHERE p.id_professor = d.id_professor);

k) SELECT *
FROM disciplina
WHERE id_professor IS NULL;

l) SELECT d.nome_disciplina,
COUNT(*) AS quantidade_alunos
FROM disciplina AS d,
matriculado AS m
WHERE d.id_disciplina = m.id_disciplina
GROUP BY d.nome_disciplina;

m) SELECT p.nome_professor,
SUM(d.carga_horaria) AS CH
FROM professor AS p,
disciplina AS d
WHERE p.id_professor = d.id_professor
GROUP BY p.nome_professor;

n) SELECT a.nome_aluno,
AVG(m.nota) AS media
FROM aluno AS a,
```

matriculado AS m
WHERE a.id_aluno = m.id_aluno
GROUP BY a.nome_aluno
ORDER BY media DESC;

o) SELECT a.nome_aluno
FROM aluno AS a,
disciplina AS d,
matriculado AS m
WHERE a.id_aluno = m.id_aluno
AND d.id_disciplina = m.id_disciplina
AND d.carga_horaria >= 60;



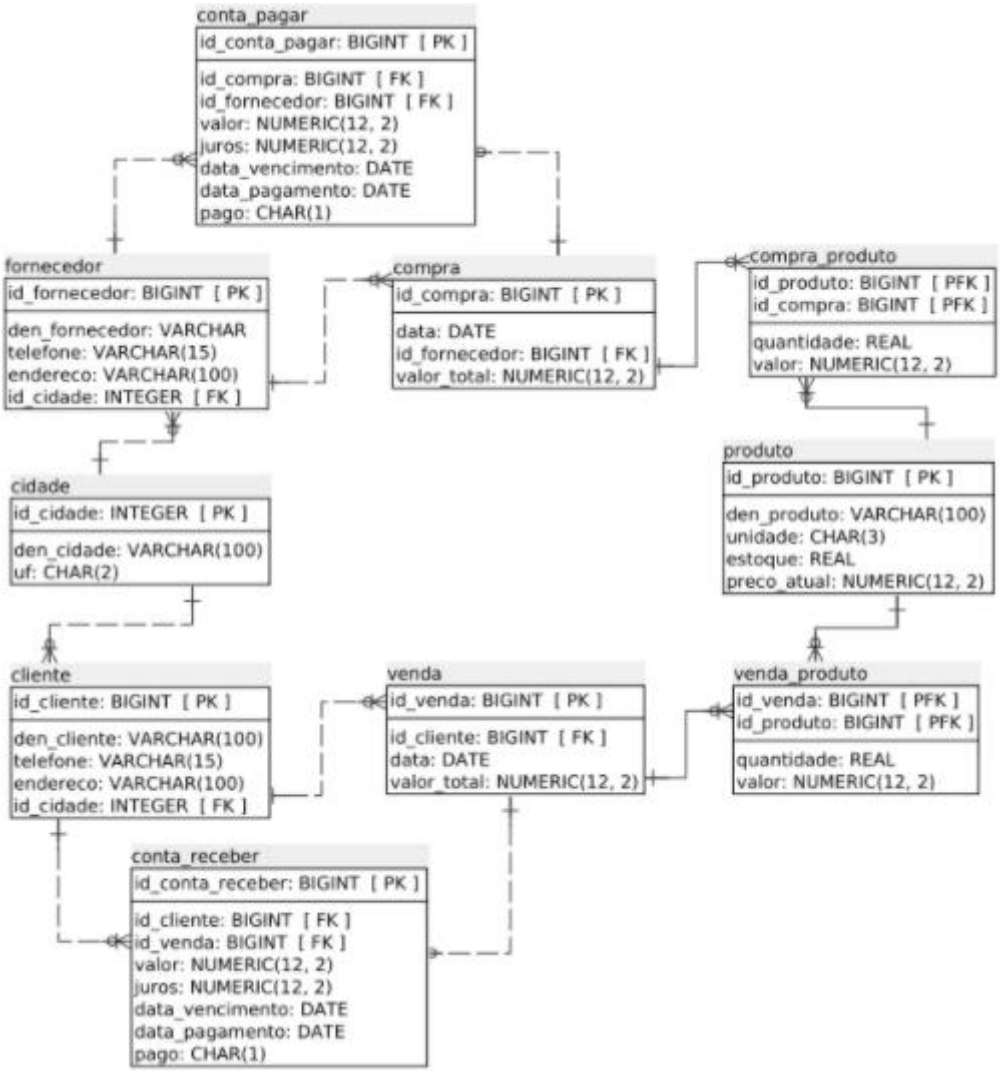
Exercício 2:
Considere o banco de dados da Figura 1. Considere também que as notas iguais a -1 (menos um) na tabela matriculado indicam que o aluno está apenas matriculado, mas não possui nenhuma nota. Escreva as instruções SQL para executar as seguintes ações:

(a) Crie as regras necessárias para que a média das disciplinas seja atualizada automaticamente (lembre-se de desconsiderar as matrículas sem notas);

CREATE OR REPLACE RULE matriculado_insert AS
ON INSERT TO matriculado DO ALSO (
UPDATE disciplina AS d
SET media = am.media
FROM (
SELECT id_disciplina,
AVG(nota) AS media
FROM matriculado AS m
GROUP BY id_disciplina) AS am
WHERE d.id_disciplina = am.id_ disciplina
AND d.id_ disciplina = NEW.id_ disciplina;
);

(b) Matricular os novos alunos automaticamente nas disciplinas sem pré-requisitos;

```
CREATE OR REPLACE RULE id_aluno AS
ON UPDATE TO matriculado
WHERE nota = 0;
DO INSTEAD NOTHING
```



Exercício 3:
Considere o banco de dados de uma empresa de varejo cujo esquema lógico é apresentado na Figura 2. Escreva as instruções SQL para executar as seguintes ações:
(a)Atualizar automaticamente os estoques dos produtos de acordo com as compras e vendas;

```
CREATE OR REPLACE VIEW estoque AS
SELECT id_compra FROM compra;
CREATE TABLE compra2 (id_compra INTEGER);
CREATE RULE _RETURN AS
ON SELECT TO compra2 DO INSTEAD (
SELECT id_compra FROM compra;
```

);

(b) Atualizar automaticamente o total das vendas de acordo com os itens vendidos;

```
CREATE OR REPLACE VIEW estoque AS
SELECT vendas FROM compra;
CREATE TABLE compra2 (vendas INTEGER);
CREATE RULE _RETURN AS
ON SELECT TO compra2 DO INSTEAD (
SELECT vendas FROM compra;
);
```

(c) Atualizar automaticamente o total das compras de acordo com os itens comprados;

```
CREATE OR REPLACE VIEW total AS
SELECT id_compra FROM compra;
CREATE TABLE ttotal (id_compra INTEGER);
CREATE RULE _RETURN AS
ON SELECT TO total DO INSTEAD (
SELECT id_compra FROM compra;
);
```

(d) Crie os atributos de limite de crédito e saldo de crédito para os clientes. Atributa o valor de 30% do total de vendas de cada cliente para seu limite de crédito e para seu saldo de crédito. Faça com que o saldo de crédito seja atualizado automaticamente de acordo com as vendas.

```
ALTER TABLE cliente ADD limite_credito FLOAT;
ALTER TABLE cliente ADD saldo_credito FLOAT;
UPDATE INTO cliente
(id_cliente, den_cliente, telefone, endereço, id_cidade, limite_credito, saldo_credito)
SELECT c.id_cliente, c.den_cliente, c.telefone, c.endereco, c.id_cidade, 0.30 *
v.valor_total, 0.30 * v.valor_total
FROM cliente AS c, venda AS v
WHERE c.id_cliente = v.id_cliente;
```