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,

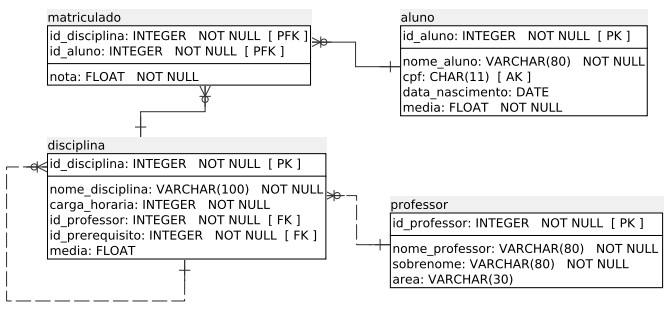
disciplina AS d,

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;

);

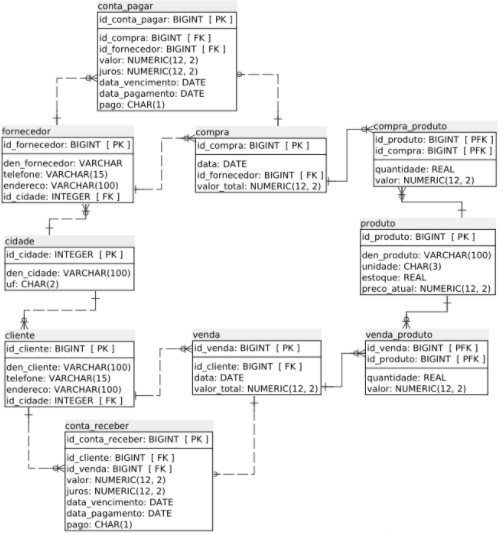
(b) Matricular os novos alunos automaticamente nas disciplinas sem pré-requisitos;

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;