

**INSTITUTO FEDERAL MINAS GERAIS (IFMG) - CAMPUS BAMBUÍ**

**Banco de Dados II**

**Prof. Marcos Roberto Ribeiro**

**Lista de Exercícios 5**

Beatriz Rodrigues de Oliveira Paiva



Figura 1: Banco de dados acadêmico

# Exercício 1:

Considere agora o banco de dados *Acadêmico* mostrado na Figura [1.](#_bookmark0) Escreva as instruções SQL para executar as seguintes ações:

1. Obter o nome do aluno, o nome da disciplina e a nota obtida pelo aluno na disciplina;

select a.nome\_aluno,

d.nome\_disciplina,

m.nota

from aluno as a,

disciplina as d,

matriculado as m

where a.id\_aluno = m.id\_aluno and d.id\_disciplina = m.id\_disciplina

ORDER BY (a.id\_aluno);

1. Obter a quantidade de disciplinas ministradas por cada professor;

select p.nome\_professor,

count (\*) as num\_disciplinas

from professor as p,

disciplina as d

where p.id\_professor = d.id\_professor

GROUP BY p.id\_professor

UNION

select distinct p.nome\_professor,

0

from professor as p,

disciplina as d

where not exists(

select d.id\_professor

from disciplina as d

where p.id\_professor = d.id\_professor

);

1. Obter os nomes completos de todos os professores com suas carga horária total;

select p.nome\_professor ||' '|| p.sobrenome as Nome\_Completo,

SUM(d.carga\_horaria) as carga\_total

from professor as p,

disciplina as d

where p.id\_professor = d.id\_professor

GROUP BY p.id\_professor

UNION

select p.nome\_professor ||' '|| p.sobrenome,

0

FROM professor as p,

disciplina as d

WHERE NOT EXISTS (

SELECT d.id\_professor

FROM disciplina AS d

WHERE p.id\_professor = d.id\_professor);

1. Obter a nota média para cada disciplina;

select d.nome\_disciplina,

AVG(m.nota)

from disciplina as d,

matriculado as m

where d.id\_disciplina = m.id\_disciplina

GROUP BY d.id\_disciplina;

1. Obter a maior e a menor nota para cada uma das disciplinas;

select d.nome\_disciplina,

MAX(m.nota),

MIN(m.nota)

FROM disciplina as d,

matriculado as m

where d.id\_disciplina = m.id\_disciplina

GROUP BY d.id\_disciplina;

1. Obter as disciplinas que o aluno *José* está matriculado e que possuam pelo menos 2 alunos matri- culados.

select d.nome\_disciplina

from disciplina as d,

matriculado as m

where (d.id\_disciplina = m.id\_disciplina)

and (m.id\_aluno = (select id\_aluno from aluno where nome\_aluno='José'))

INTERSECT

select d.nome\_disciplina

from disciplina as d,

matriculado as m

where d.id\_disciplina = m.id\_disciplina

GROUP BY d.id\_disciplina HAVING count(\*) >=2;

1. Obter os alunos matriculados nas disciplinas com com carga horária maior ou igual a 60;

select distinct a.nome\_aluno

from aluno as a,

matriculado as m,

disciplina as d

where (a.id\_aluno = m.id\_aluno)

and (m.id\_disciplina = d.id\_disciplina)

and (d.carga\_horaria >=60);

1. Obter a média das notas de cada aluno em ordem decrescente pela média;

select nome\_aluno,

media

from aluno

ORDER BY media DESC;

1. Obter os nomes dos alunos matriculado em disciplinas de professores da área de *Computação*;

CREATE TEMPORARY TABLE media AS

SELECT id\_aluno,

AVG(nota) AS media

FROM matriculado

GROUP BY id\_aluno;

UPDATE aluno AS a

SET media = m.media

FROM media AS m WHERE a.id\_aluno = m.id\_aluno;

1. Obter a carga horária total de cada professor de acordo com as disciplinas ministradas;

select distinct a.nome\_aluno

from aluno as a,

matriculado as m,

disciplina as d

where (m.id\_aluno = a.id\_aluno)

and (d.id\_disciplina = m.id\_disciplina)

and (d.id\_professor IN (

select p.id\_professor

from professor as p

where p.area ='Computação'));

1. Obter a quantidade de alunos matriculados em cada disciplinas;

select p.nome\_professor,

SUM(d.carga\_horaria) as carga\_total

from professor as p,

disciplina as d

where p.id\_professor = d.id\_professor

GROUP BY p.id\_professor

UNION

select p.nome\_professor,

0

FROM professor as p,

disciplina as d

WHERE NOT EXISTS (

SELECT d.id\_professor

FROM disciplina AS d

WHERE p.id\_professor = d.id\_professor);

1. Obter os alunos que cursaram mais de 100 horas de disciplinas;

select d.nome\_disciplina,

count(\*)

from disciplina as d,

matriculado as m

where (d.id\_disciplina = m.id\_disciplina)

GROUP BY (d.nome\_disciplina);

1. Obter o nome dos alunos, nome das disciplinas, nome dos professores da disciplinas e a nota obtida pelos alunos;

select a.nome\_aluno

from disciplina as d,

matriculado as m,

aluno as a

where (m.id\_aluno=a.id\_aluno)

and (m.id\_disciplina = d.id\_disciplina)

GROUP BY (a.id\_aluno)

HAVING (sum(d.carga\_horaria)>100);

1. Obter as disciplinas sem professor;

select distinct a.nome\_aluno,

d.nome\_disciplina,

p.nome\_professor,

m.nota

from aluno as a,

disciplina as d,

professor as p,

matriculado as m

where (m.id\_aluno = a.id\_aluno)

and (m.id\_disciplina = d.id\_disciplina)

and (p.id\_professor = d.id\_professor)

ORDER BY (nome\_aluno);

1. Obter os professores sem disciplina;

select d.nome\_disciplina

from disciplina as d

where d.id\_professor is null;

1. Obter possíveis duplas combinando todos os nomes de alunos, mas sem combinar um aluno com ele mesmo;

select distinct p.nome\_professor

FROM professor as p,

disciplina as d

WHERE NOT EXISTS (

SELECT d.id\_professor

FROM disciplina AS d

WHERE p.id\_professor = d.id\_professor);

1. Obter as disciplinas sem nenhum aluno matriculado;

select a1.nome\_aluno, a2.nome\_aluno

from aluno as a1, aluno as a2

where a1.id\_aluno<a2.id\_aluno

order by a1.id\_aluno;

1. Matricular todos os alunos nas disciplinas sem nenhum aluno matriculado.

select d.id\_disciplina,

d.nome\_disciplina

from disciplina as d,

matriculado as m

where not exists(

select m.id\_disciplina

from matriculado as m

where (d.id\_disciplina = m.id\_disciplina))

GROUP BY (d.id\_disciplina);



Figura 2: Banco de dados de uma empresa de varejo

# Exercício 2:

Considere o banco de dados de uma empresa de varejo cujo esquema lógico é apresentado na Figura [2.](#_bookmark1) Escreva as instruções SQL para executar as seguintes ações:

1. Obter o valor total comprado de cada fornecedor;

select f.den\_fornecedor, SUM(c.valor\_total)

from fornecedor as f,

compra as c

where (c.id\_fornecedor = f.id\_fornecedor)

GROUP BY (f.id\_fornecedor);

1. O valor total vendido para cada cidade;

select ci.id\_cidade,

ci.den\_cidade,

SUM(v.valor\_total)

from cliente as c,

venda as v,

cidade as ci

where (v.id\_cliente = c.id\_cliente)

and (c.id\_cidade = ci.id\_cidade)

GROUP BY(ci.id\_cidade);

1. O valor total de cada produto vendido para cada cidade;

select ci.den\_cidade,

p.den\_produto,

SUM(vp.quantidade \* vp.valor)

from cidade as ci,

venda\_produto as vp,

cliente as c,

venda as v,

produto as p

where (vp.id\_produto = p.id\_produto)

and (vp.id\_venda = v.id\_venda)

and (v.id\_cliente = c.id\_cliente)

and (c.id\_cidade = ci.id\_cidade)

GROUP BY(ci.id\_cidade, p.id\_produto)

ORDER BY (ci.id\_cidade);

1. A quantidade, o valor total e o valor médio de cada produto comprado de cada estado (UF);

select p.id\_produto,

p.den\_produto,

ci.uf,

sum(cp.quantidade) as quantidade,

sum(cp.valor \* cp.quantidade) as total,

avg(cp.valor) as valor\_medio

from cidade as ci,

fornecedor as f,

compra as co,

produto as p,

compra\_produto as cp

where co.id\_fornecedor = f.id\_fornecedor

and f.id\_cidade = ci.id\_cidade

and co.id\_compra = cp.id\_compra

and cp.id\_produto = p.id\_produto

group by p.id\_produto, ci.uf

order by ci.uf;

1. Listar as cidades com suas respectivas quantidades de cadastros (um cadastro pode ser um cliente ou um fornecedor);

create temporary table cf as(

select id\_cidade from fornecedor

union all

select id\_cidade from cliente

);

select c.id\_cidade,

c.den\_cidade,

count(\*)as cadastros

from cidade as c, cf

where c.id\_cidade = cf.id\_cidade

group by c.id\_cidade;

1. Listar as contas a pagar vencidas até dezembro de 2011 e que não foram pagas;

select id\_conta\_pagar

from conta\_pagar

where (data\_vencimento <= '2011-12-31')

and pago <>'S';

1. Listar os fornecedores que possuem mais de 10 contas a pagar;

select f.id\_fornecedor,

f.den\_fornecedor,

count(\*) as contas\_a\_pagar

from fornecedor as f,

conta\_pagar as cp

where f.id\_fornecedor = cp.id\_fornecedor

GROUP BY f.id\_fornecedor

HAVING count(\*)>10;

1. Listar o total devido por cada cliente;

select \* from((select c.id\_cliente,

SUM(cr.valor) as valor

from cliente as c,

conta\_receber as cr

where (c.id\_cliente = cr.id\_cliente)

and (cr.pago = 'N')

GROUP BY (c.id\_cliente)

ORDER BY (c.id\_cliente))

UNION ALL

((select id\_cliente, 0

from cliente)

EXCEPT ALL

(select c.id\_cliente, 0

from cliente as c,

conta\_receber as cr

where (c.id\_cliente = cr.id\_cliente)

and (cr.pago = 'N')

GROUP BY (c.id\_cliente))))

as consulta

ORDER BY (consulta.id\_cliente);

1. Listar os 10 produtos com maior movimentação (considerando compras e vendas);

create temporary table movimentacao as (

select id\_produto,

quantidade

from venda\_produto

union all

select id\_produto,

quantidade

from compra\_produto

);

select p.id\_produto,

p.den\_produto,

sum(m.quantidade) as quantidade

from produto as p,

movimentacao as m

where p.id\_produto = m.id\_produto

group by (p.id\_produto)

order by quantidade DESC

limit 10;

1. Listar o faturamento (vendas) mensal de todos os meses;

SELECT EXTRACT(year FROM data) || '-' || DATE\_PART('month', data) AS mes,

SUM(vp.quantidade) AS Faturamento\_mensal

FROM venda AS v,

venda\_produto AS vp

WHERE v.id\_venda = vp.id\_venda

GROUP BY EXTRACT(year FROM data) || '-' || DATE\_PART('month', data);

1. Listar os produtos que foram vendidos, mas não foram comprados em janeiro de 2011;

CREATE TEMP TABLE media AS

WITH mm as (

SELECT vp.id\_produto,

EXTRACT(year FROM data) || '-' || DATE\_PART('month', data) AS mes,

AVG(vp.quantidade) AS media

FROM venda AS v,

venda\_produto AS vp

WHERE v.id\_venda = vp.id\_venda

GROUP BY id\_produto, EXTRACT(year FROM data) || '-' || DATE\_PART('month', data)

)

SELECT id\_produto,

AVG(media) AS media

FROM mm

GROUP BY id\_produto;

ALTER TABLE produto ADD estoque\_minimo REAL;

UPDATE produto AS p

SET estoque\_minimo = m.media \* 0.5

FROM media AS m

WHERE m.id\_produto = p.id\_produto;

1. Listar os produtos que foram comprados em 2010, sem repetições;

select distinct p.id\_produto

from produto as p,

venda\_produto as vp

where (p.id\_produto = vp.id\_produto)

EXCEPT

select distinct p.id\_produto

from produto as p,

compra\_produto as cp,

compra as c

where (p.id\_produto = cp.id\_produto)

and (c.id\_compra = cp.id\_compra)

and (c.data >='2011-01-01')

and (c.data <='2011-01-31');

1. Listar as vendas de 2011 contendo mais de 5 produtos que ainda não foram pagas;

select distinct p.den\_produto

from produto as p,

compra\_produto as cp,

compra as c

where (p.id\_produto = cp.id\_produto)

and (cp.id\_compra = c.id\_compra)

and (c.data>='2010-01-01')

and (c.data<='2010-12-31');

1. Listar o total vendido e o total comprado de cada produto (pode acontecer de um produto ter sido comprado e não ter sido vendido e vice-versa);

select v.id\_venda

from produto as p,

venda\_produto as vp,

venda as v,

conta\_receber as cr

where (v.id\_venda = vp.id\_venda)

and (vp.id\_produto = p.id\_produto)

and (v.data >= '2011-01-01')

and (v.data <= '2011-12-31')

and (cr.id\_venda = v.id\_venda)

GROUP BY (v.id\_venda)

HAVING (count(\*)>=5);

1. Listar o mês e ano com a maior quantidade de vendas de cada produto.

select p.den\_produto,

(select SUM(vp.quantidade) as total\_vendido

from venda\_produto as vp

where (p.id\_produto = vp.id\_produto)

GROUP BY (p.id\_produto)),

(select SUM(cp.quantidade) as total\_comprado

from compra\_produto as cp

where (p.id\_produto = cp.id\_produto)

GROUP BY (p.id\_produto))

from produto as p;