-- =====================================================

-- CONSULTAS DE MÉTRICAS PARA E-COMMERCE

-- =====================================================

-- =====================================================

-- 1. MÉTRICAS DE VENDAS E FATURAMENTO

-- =====================================================

-- 1.1 Faturamento Total por Período

SELECT

DATE\_FORMAT(data\_pedido, '%Y-%m') as periodo,

COUNT(\*) as total\_pedidos,

SUM(valor\_total) as faturamento\_total,

AVG(valor\_total) as ticket\_medio,

SUM(valor\_frete) as total\_frete

FROM Pedido p

JOIN Status\_Pedido sp ON p.idStatus = sp.idStatus

WHERE sp.status\_final = TRUE

AND sp.nome\_status != 'Cancelado'

AND data\_pedido >= DATE\_SUB(CURDATE(), INTERVAL 12 MONTH)

GROUP BY DATE\_FORMAT(data\_pedido, '%Y-%m')

ORDER BY periodo DESC;

-- 1.2 Faturamento por Dia da Semana

SELECT

DAYNAME(data\_pedido) as dia\_semana,

DAYOFWEEK(data\_pedido) as numero\_dia,

COUNT(\*) as total\_pedidos,

SUM(valor\_total) as faturamento,

AVG(valor\_total) as ticket\_medio

FROM Pedido p

JOIN Status\_Pedido sp ON p.idStatus = sp.idStatus

WHERE sp.status\_final = TRUE

AND sp.nome\_status != 'Cancelado'

AND data\_pedido >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

GROUP BY DAYOFWEEK(data\_pedido), DAYNAME(data\_pedido)

ORDER BY numero\_dia;

-- 1.3 Top 10 Produtos Mais Vendidos

SELECT

pr.nome as produto,

pr.sku,

cat.nome as categoria,

SUM(rpp.quantidade) as total\_vendido,

COUNT(DISTINCT rpp.idPedido) as pedidos\_diferentes,

SUM(rpp.preco\_total) as faturamento\_produto,

AVG(rpp.preco\_unitario) as preco\_medio

FROM Relacao\_Produto\_Pedido rpp

JOIN Produto pr ON rpp.idProduto = pr.idProduto

JOIN Categoria cat ON pr.idCategoria = cat.idCategoria

JOIN Pedido p ON rpp.idPedido = p.idPedido

JOIN Status\_Pedido sp ON p.idStatus = sp.idStatus

WHERE sp.status\_final = TRUE

AND sp.nome\_status != 'Cancelado'

AND p.data\_pedido >= DATE\_SUB(CURDATE(), INTERVAL 6 MONTH)

GROUP BY pr.idProduto, pr.nome, pr.sku, cat.nome

ORDER BY total\_vendido DESC

LIMIT 10;

-- =====================================================

-- 2. MÉTRICAS DE CLIENTES

-- =====================================================

-- 2.1 Análise de Clientes por Tipo (PF vs PJ)

SELECT

c.tipo\_cliente,

COUNT(\*) as total\_clientes,

COUNT(CASE WHEN p.idPedido IS NOT NULL THEN 1 END) as clientes\_compraram,

ROUND(COUNT(CASE WHEN p.idPedido IS NOT NULL THEN 1 END) \* 100.0 / COUNT(\*), 2) as taxa\_conversao\_pct,

AVG(pedidos\_por\_cliente.total\_pedidos) as media\_pedidos\_por\_cliente,

AVG(pedidos\_por\_cliente.valor\_total\_cliente) as valor\_medio\_por\_cliente

FROM Cliente c

LEFT JOIN (

SELECT

idCliente,

COUNT(\*) as total\_pedidos,

SUM(valor\_total) as valor\_total\_cliente

FROM Pedido p2

JOIN Status\_Pedido sp2 ON p2.idStatus = sp2.idStatus

WHERE sp2.status\_final = TRUE AND sp2.nome\_status != 'Cancelado'

GROUP BY idCliente

) pedidos\_por\_cliente ON c.idCliente = pedidos\_por\_cliente.idCliente

LEFT JOIN Pedido p ON c.idCliente = p.idCliente

GROUP BY c.tipo\_cliente;

-- 2.2 Top 10 Melhores Clientes (por valor total)

SELECT

c.nome,

c.email,

c.tipo\_cliente,

COUNT(p.idPedido) as total\_pedidos,

SUM(p.valor\_total) as valor\_total\_gasto,

AVG(p.valor\_total) as ticket\_medio,

MAX(p.data\_pedido) as ultima\_compra,

DATEDIFF(CURDATE(), MAX(p.data\_pedido)) as dias\_sem\_comprar

FROM Cliente c

JOIN Pedido p ON c.idCliente = p.idCliente

JOIN Status\_Pedido sp ON p.idStatus = sp.idStatus

WHERE sp.status\_final = TRUE

AND sp.nome\_status != 'Cancelado'

GROUP BY c.idCliente, c.nome, c.email, c.tipo\_cliente

ORDER BY valor\_total\_gasto DESC

LIMIT 10;

-- 2.3 Clientes Inativos (não compraram nos últimos 90 dias)

SELECT

c.nome,

c.email,

c.tipo\_cliente,

COUNT(p.idPedido) as total\_pedidos\_historico,

SUM(p.valor\_total) as valor\_total\_historico,

MAX(p.data\_pedido) as ultima\_compra,

DATEDIFF(CURDATE(), MAX(p.data\_pedido)) as dias\_sem\_comprar

FROM Cliente c

JOIN Pedido p ON c.idCliente = p.idCliente

JOIN Status\_Pedido sp ON p.idStatus = sp.idStatus

WHERE sp.status\_final = TRUE

AND sp.nome\_status != 'Cancelado'

GROUP BY c.idCliente, c.nome, c.email, c.tipo\_cliente

HAVING MAX(p.data\_pedido) < DATE\_SUB(CURDATE(), INTERVAL 90 DAY)

ORDER BY valor\_total\_historico DESC;

-- =====================================================

-- 3. MÉTRICAS DE ESTOQUE

-- =====================================================

-- 3.1 Produtos com Estoque Baixo

SELECT

pr.nome,

pr.sku,

cat.nome as categoria,

est.local as estoque\_local,

pe.quantidade\_atual,

pe.quantidade\_minima,

pe.quantidade\_reservada,

(pe.quantidade\_atual - pe.quantidade\_reservada) as disponivel\_venda,

CASE

WHEN pe.quantidade\_atual <= pe.quantidade\_minima THEN 'CRÍTICO'

WHEN pe.quantidade\_atual <= (pe.quantidade\_minima \* 1.5) THEN 'BAIXO'

ELSE 'OK'

END as status\_estoque

FROM Produto pr

JOIN Categoria cat ON pr.idCategoria = cat.idCategoria

JOIN Produto\_Estoque pe ON pr.idProduto = pe.idProduto

JOIN Estoque est ON pe.idEstoque = est.idEstoque

WHERE pr.ativo = TRUE

AND pe.quantidade\_atual <= (pe.quantidade\_minima \* 1.5)

ORDER BY

CASE

WHEN pe.quantidade\_atual <= pe.quantidade\_minima THEN 1

ELSE 2

END,

pe.quantidade\_atual ASC;

-- 3.2 Giro de Estoque por Produto (últimos 6 meses)

SELECT

pr.nome,

pr.sku,

cat.nome as categoria,

SUM(pe.quantidade\_atual) as estoque\_atual\_total,

COALESCE(SUM(rpp.quantidade), 0) as vendido\_6\_meses,

CASE

WHEN SUM(pe.quantidade\_atual) > 0 THEN

ROUND(COALESCE(SUM(rpp.quantidade), 0) / SUM(pe.quantidade\_atual), 2)

ELSE 0

END as giro\_estoque,

CASE

WHEN COALESCE(SUM(rpp.quantidade), 0) = 0 THEN 'SEM MOVIMENTO'

WHEN SUM(pe.quantidade\_atual) = 0 THEN 'SEM ESTOQUE'

WHEN COALESCE(SUM(rpp.quantidade), 0) / SUM(pe.quantidade\_atual) > 2 THEN 'ALTO GIRO'

WHEN COALESCE(SUM(rpp.quantidade), 0) / SUM(pe.quantidade\_atual) > 1 THEN 'GIRO BOM'

ELSE 'GIRO BAIXO'

END as classificacao\_giro

FROM Produto pr

JOIN Categoria cat ON pr.idCategoria = cat.idCategoria

JOIN Produto\_Estoque pe ON pr.idProduto = pe.idProduto

LEFT JOIN Relacao\_Produto\_Pedido rpp ON pr.idProduto = rpp.idProduto

LEFT JOIN Pedido p ON rpp.idPedido = p.idPedido AND p.data\_pedido >= DATE\_SUB(CURDATE(), INTERVAL 6 MONTH)

LEFT JOIN Status\_Pedido sp ON p.idStatus = sp.idStatus AND sp.status\_final = TRUE AND sp.nome\_status != 'Cancelado'

WHERE pr.ativo = TRUE

GROUP BY pr.idProduto, pr.nome, pr.sku, cat.nome

ORDER BY giro\_estoque DESC;

-- =====================================================

-- 4. MÉTRICAS DE PERFORMANCE POR CATEGORIA

-- =====================================================

-- 4.1 Performance de Vendas por Categoria

SELECT

cat.nome as categoria,

COUNT(DISTINCT pr.idProduto) as total\_produtos,

COUNT(DISTINCT p.idPedido) as total\_pedidos,

SUM(rpp.quantidade) as total\_itens\_vendidos,

SUM(rpp.preco\_total) as faturamento\_categoria,

AVG(rpp.preco\_unitario) as preco\_medio,

ROUND(SUM(rpp.preco\_total) \* 100.0 / (

SELECT SUM(rpp2.preco\_total)

FROM Relacao\_Produto\_Pedido rpp2

JOIN Pedido p2 ON rpp2.idPedido = p2.idPedido

JOIN Status\_Pedido sp2 ON p2.idStatus = sp2.idStatus

WHERE sp2.status\_final = TRUE

AND sp2.nome\_status != 'Cancelado'

AND p2.data\_pedido >= DATE\_SUB(CURDATE(), INTERVAL 6 MONTH)

), 2) as participacao\_faturamento\_pct

FROM Categoria cat

JOIN Produto pr ON cat.idCategoria = pr.idCategoria

JOIN Relacao\_Produto\_Pedido rpp ON pr.idProduto = rpp.idProduto

JOIN Pedido p ON rpp.idPedido = p.idPedido

JOIN Status\_Pedido sp ON p.idStatus = sp.idStatus

WHERE sp.status\_final = TRUE

AND sp.nome\_status != 'Cancelado'

AND p.data\_pedido >= DATE\_SUB(CURDATE(), INTERVAL 6 MONTH)

AND pr.ativo = TRUE

GROUP BY cat.idCategoria, cat.nome

ORDER BY faturamento\_categoria DESC;

-- =====================================================

-- 5. MÉTRICAS DE ENTREGA E LOGÍSTICA

-- =====================================================

-- 5.1 Performance de Entregas

SELECT

e.status\_entrega,

COUNT(\*) as total\_entregas,

ROUND(COUNT(\*) \* 100.0 / (SELECT COUNT(\*) FROM Entrega), 2) as percentual,

AVG(DATEDIFF(e.data\_entrega\_real, e.data\_despacho)) as media\_dias\_entrega,

AVG(DATEDIFF(e.data\_previsao\_entrega, e.data\_despacho)) as media\_prazo\_previsto,

AVG(e.custo\_entrega) as custo\_medio\_entrega

FROM Entrega e

WHERE e.data\_despacho >= DATE\_SUB(CURDATE(), INTERVAL 3 MONTH)

GROUP BY e.status\_entrega

ORDER BY total\_entregas DESC;

-- 5.2 Entregas em Atraso

SELECT

p.numero\_pedido,

c.nome as cliente,

e.data\_previsao\_entrega,

e.data\_entrega\_real,

DATEDIFF(COALESCE(e.data\_entrega\_real, CURDATE()), e.data\_previsao\_entrega) as dias\_atraso,

e.status\_entrega,

e.nome\_transportadora

FROM Entrega e

JOIN Pedido p ON e.idPedido = p.idPedido

JOIN Cliente c ON p.idCliente = c.idCliente

WHERE (

(e.data\_entrega\_real IS NULL AND CURDATE() > e.data\_previsao\_entrega)

OR

(e.data\_entrega\_real IS NOT NULL AND e.data\_entrega\_real > e.data\_previsao\_entrega)

)

AND e.data\_previsao\_entrega >= DATE\_SUB(CURDATE(), INTERVAL 1 MONTH)

ORDER BY dias\_atraso DESC;

-- =====================================================

-- 6. MÉTRICAS DE FORMA DE PAGAMENTO

-- =====================================================

-- 6.1 Preferências de Pagamento

SELECT

fp.nome as forma\_pagamento,

fp.tipo\_pagamento,

COUNT(p.idPedido) as total\_pedidos,

SUM(p.valor\_total) as faturamento\_total,

ROUND(COUNT(p.idPedido) \* 100.0 / (

SELECT COUNT(\*) FROM Pedido p2

JOIN Status\_Pedido sp2 ON p2.idStatus = sp2.idStatus

WHERE sp2.status\_final = TRUE AND sp2.nome\_status != 'Cancelado'

), 2) as percentual\_uso,

AVG(p.valor\_total) as ticket\_medio

FROM Forma\_Pagamento fp

JOIN Pedido p ON fp.idFormaPagamento = p.idFormaPagamento

JOIN Status\_Pedido sp ON p.idStatus = sp.idStatus

WHERE sp.status\_final = TRUE

AND sp.nome\_status != 'Cancelado'

AND p.data\_pedido >= DATE\_SUB(CURDATE(), INTERVAL 6 MONTH)

GROUP BY fp.idFormaPagamento, fp.nome, fp.tipo\_pagamento

ORDER BY total\_pedidos DESC;

-- =====================================================

-- 7. DASHBOARD EXECUTIVO (RESUMO GERAL)

-- =====================================================

-- 7.1 KPIs Principais do Mês Atual

SELECT

'VENDAS\_MES\_ATUAL' as metrica,

COUNT(\*) as valor,

'pedidos' as unidade

FROM Pedido p

JOIN Status\_Pedido sp ON p.idStatus = sp.idStatus

WHERE sp.status\_final = TRUE

AND sp.nome\_status != 'Cancelado'

AND MONTH(p.data\_pedido) = MONTH(CURDATE())

AND YEAR(p.data\_pedido) = YEAR(CURDATE())

UNION ALL

SELECT

'FATURAMENTO\_MES\_ATUAL',

ROUND(SUM(p.valor\_total), 2),

'reais'

FROM Pedido p

JOIN Status\_Pedido sp ON p.idStatus = sp.idStatus

WHERE sp.status\_final = TRUE

AND sp.nome\_status != 'Cancelado'

AND MONTH(p.data\_pedido) = MONTH(CURDATE())

AND YEAR(p.data\_pedido) = YEAR(CURDATE())

UNION ALL

SELECT

'TICKET\_MEDIO\_MES\_ATUAL',

ROUND(AVG(p.valor\_total), 2),

'reais'

FROM Pedido p

JOIN Status\_Pedido sp ON p.idStatus = sp.idStatus

WHERE sp.status\_final = TRUE

AND sp.nome\_status != 'Cancelado'

AND MONTH(p.data\_pedido) = MONTH(CURDATE())

AND YEAR(p.data\_pedido) = YEAR(CURDATE())

UNION ALL

SELECT

'NOVOS\_CLIENTES\_MES\_ATUAL',

COUNT(\*),

'clientes'

FROM Cliente c

WHERE MONTH(c.data\_cadastro) = MONTH(CURDATE())

AND YEAR(c.data\_cadastro) = YEAR(CURDATE())

UNION ALL

SELECT

'PRODUTOS\_ESTOQUE\_CRITICO',

COUNT(\*),

'produtos'

FROM Produto\_Estoque pe

JOIN Produto pr ON pe.idProduto = pr.idProduto

WHERE pr.ativo = TRUE

AND pe.quantidade\_atual <= pe.quantidade\_minima;

-- =====================================================

-- 8. ANÁLISE DE TENDÊNCIAS (CRESCIMENTO MÊS A MÊS)

-- =====================================================

-- 8.1 Crescimento de Vendas (Últimos 12 meses)

SELECT

DATE\_FORMAT(p.data\_pedido, '%Y-%m') as mes,

COUNT(\*) as pedidos\_mes,

SUM(p.valor\_total) as faturamento\_mes,

LAG(COUNT(\*)) OVER (ORDER BY DATE\_FORMAT(p.data\_pedido, '%Y-%m')) as pedidos\_mes\_anterior,

LAG(SUM(p.valor\_total)) OVER (ORDER BY DATE\_FORMAT(p.data\_pedido, '%Y-%m')) as faturamento\_mes\_anterior,

ROUND(

(COUNT(\*) - LAG(COUNT(\*)) OVER (ORDER BY DATE\_FORMAT(p.data\_pedido, '%Y-%m'))) \* 100.0

/ LAG(COUNT(\*)) OVER (ORDER BY DATE\_FORMAT(p.data\_pedido, '%Y-%m')), 2

) as crescimento\_pedidos\_pct,

ROUND(

(SUM(p.valor\_total) - LAG(SUM(p.valor\_total)) OVER (ORDER BY DATE\_FORMAT(p.data\_pedido, '%Y-%m'))) \* 100.0

/ LAG(SUM(p.valor\_total)) OVER (ORDER BY DATE\_FORMAT(p.data\_pedido, '%Y-%m')), 2

) as crescimento\_faturamento\_pct

FROM Pedido p

JOIN Status\_Pedido sp ON p.idStatus = sp.idStatus

WHERE sp.status\_final = TRUE

AND sp.nome\_status != 'Cancelado'

AND p.data\_pedido >= DATE\_SUB(CURDATE(), INTERVAL 12 MONTH)

GROUP BY DATE\_FORMAT(p.data\_pedido, '%Y-%m')

ORDER BY mes DESC;

-- =====================================================

-- 9. CONSULTAS ESPECÍFICAS SOLICITADAS

-- =====================================================

-- 9.1 QUANTOS PEDIDOS FORAM FEITOS POR CADA CLIENTE?

SELECT

c.idCliente,

c.nome as nome\_cliente,

c.email,

c.tipo\_cliente,

COUNT(p.idPedido) as total\_pedidos,

COUNT(CASE WHEN sp.status\_final = TRUE AND sp.nome\_status != 'Cancelado' THEN 1 END) as pedidos\_concluidos,

COUNT(CASE WHEN sp.nome\_status = 'Cancelado' THEN 1 END) as pedidos\_cancelados,

COUNT(CASE WHEN sp.status\_final = FALSE THEN 1 END) as pedidos\_em\_andamento,

COALESCE(SUM(CASE WHEN sp.status\_final = TRUE AND sp.nome\_status != 'Cancelado' THEN p.valor\_total END), 0) as valor\_total\_gasto,

MIN(p.data\_pedido) as primeiro\_pedido,

MAX(p.data\_pedido) as ultimo\_pedido

FROM Cliente c

LEFT JOIN Pedido p ON c.idCliente = p.idCliente

LEFT JOIN Status\_Pedido sp ON p.idStatus = sp.idStatus

GROUP BY c.idCliente, c.nome, c.email, c.tipo\_cliente

ORDER BY total\_pedidos DESC, valor\_total\_gasto DESC;

-- 9.2 ALGUM VENDEDOR TAMBÉM É FORNECEDOR?

-- Verificando por CNPJ (chave de identificação empresarial)

SELECT

tv.idTerceiro,

tv.razao\_social as razao\_social\_vendedor,

tv.cnpj as cnpj\_vendedor,

f.idFornecedor,

f.razao\_social as razao\_social\_fornecedor,

f.cnpj as cnpj\_fornecedor,

'CNPJ\_IDENTICO' as tipo\_match

FROM Terceiro\_Vendedor tv

INNER JOIN Fornecedor f ON tv.cnpj = f.cnpj

WHERE tv.cnpj IS NOT NULL

AND f.cnpj IS NOT NULL

AND tv.ativo = TRUE

AND f.ativo = TRUE

UNION ALL

-- Verificando por similaridade de razão social (possível mesmo grupo empresarial)

SELECT

tv.idTerceiro,

tv.razao\_social as razao\_social\_vendedor,

tv.cnpj as cnpj\_vendedor,

f.idFornecedor,

f.razao\_social as razao\_social\_fornecedor,

f.cnpj as cnpj\_fornecedor,

'RAZAO\_SOCIAL\_SIMILAR' as tipo\_match

FROM Terceiro\_Vendedor tv

INNER JOIN Fornecedor f ON UPPER(TRIM(tv.razao\_social)) = UPPER(TRIM(f.razao\_social))

WHERE tv.cnpj != f.cnpj -- CNPJs diferentes mas razão social igual

AND tv.ativo = TRUE

AND f.ativo = TRUE

ORDER BY tipo\_match, razao\_social\_vendedor;

-- 9.3 RELAÇÃO DE PRODUTOS, FORNECEDORES E ESTOQUES

SELECT

pr.idProduto,

pr.nome as nome\_produto,

pr.sku,

cat.nome as categoria,

-- Informações do Fornecedor

f.idFornecedor,

f.razao\_social as nome\_fornecedor,

f.cnpj as cnpj\_fornecedor,

dp.preco\_custo as preco\_fornecedor,

dp.data\_atualizacao as data\_atualizacao\_preco,

-- Informações do Estoque

est.idEstoque,

est.local as local\_estoque,

pe.quantidade\_atual,

pe.quantidade\_reservada,

pe.quantidade\_minima,

(pe.quantidade\_atual - pe.quantidade\_reservada) as disponivel\_venda,

pe.data\_ultima\_movimentacao,

-- Status do produto

pr.ativo as produto\_ativo,

f.ativo as fornecedor\_ativo,

est.ativo as estoque\_ativo

FROM Produto pr

JOIN Categoria cat ON pr.idCategoria = cat.idCategoria

LEFT JOIN Disponibilizar\_Produto dp ON pr.idProduto = dp.idProduto

LEFT JOIN Fornecedor f ON dp.idFornecedor = f.idFornecedor

LEFT JOIN Produto\_Estoque pe ON pr.idProduto = pe.idProduto

LEFT JOIN Estoque est ON pe.idEstoque = est.idEstoque

WHERE pr.ativo = TRUE

ORDER BY pr.nome, f.razao\_social, est.local;

-- 9.4 RELAÇÃO DE NOMES DOS FORNECEDORES E NOMES DOS PRODUTOS

SELECT

f.idFornecedor,

f.razao\_social as nome\_fornecedor,

f.cnpj,

f.telefone as telefone\_fornecedor,

f.email as email\_fornecedor,

COUNT(dp.idProduto) as total\_produtos\_fornecidos,

-- Lista dos produtos (limitada para não ficar muito extensa)

GROUP\_CONCAT(

DISTINCT

CONCAT(pr.nome, ' (SKU: ', pr.sku, ')')

ORDER BY pr.nome

SEPARATOR ' | '

) as produtos\_fornecidos,

AVG(dp.preco\_custo) as preco\_custo\_medio,

MIN(dp.data\_atualizacao) as primeiro\_produto\_cadastrado,

MAX(dp.data\_atualizacao) as ultimo\_produto\_atualizado

FROM Fornecedor f

LEFT JOIN Disponibilizar\_Produto dp ON f.idFornecedor = dp.idFornecedor

LEFT JOIN Produto pr ON dp.idProduto = pr.idProduto AND pr.ativo = TRUE

WHERE f.ativo = TRUE

GROUP BY f.idFornecedor, f.razao\_social, f.cnpj, f.telefone, f.email

ORDER BY total\_produtos\_fornecidos DESC, f.razao\_social;

-- 9.4.1 VERSÃO DETALHADA (Produto por Produto) da relação Fornecedor x Produto

SELECT

f.idFornecedor,

f.razao\_social as nome\_fornecedor,

pr.idProduto,

pr.nome as nome\_produto,

pr.sku,

cat.nome as categoria\_produto,

m.nome as marca\_produto,

dp.preco\_custo,

pr.preco\_venda,

ROUND(((pr.preco\_venda - dp.preco\_custo) / pr.preco\_venda) \* 100, 2) as margem\_lucro\_pct,

dp.data\_atualizacao as data\_ultima\_atualizacao\_preco

FROM Fornecedor f

INNER JOIN Disponibilizar\_Produto dp ON f.idFornecedor = dp.idFornecedor

INNER JOIN Produto pr ON dp.idProduto = pr.idProduto

INNER JOIN Categoria cat ON pr.idCategoria = cat.idCategoria

LEFT JOIN Marca m ON pr.idMarca = m.idMarca

WHERE f.ativo = TRUE

AND pr.ativo = TRUE

ORDER BY f.razao\_social, cat.nome, pr.nome;