LOAD CSV WITH HEADERS FROM 'file:///Supermartnew22.csv' AS row

WITH row WHERE row.CustomerName IS NOT NULL

MERGE (o:OrderID {name: row.OrderID,

CustomerName: row.CustomerName,

Gender: row.Gender,

Category: row.Category,

SubCategory: row.SubCategory,

City: row.City,

OrderDate: row.OrderDate,

Day: row.Day,

Region: row.Region,

Sales: row.Sales,

Profit: row.Profit})

MERGE (c:CustomerName {name: row.CustomerName})

MERGE (g:Gender {name: row.Gender})

MERGE (cat:Category {name: row.Category})

MERGE (subcat:SubCategory {name: row.SubCategory})

MERGE (city:City {name: row.City})

MERGE (r:Region {name: row.Region})

// İlişkiler

MERGE (o)-[:ORDERED\_BY]->(c)

MERGE (o)-[:HAS\_CATEGORY]->(cat)

MERGE (o)-[:HAS\_SUBCATEGORY]->(subcat)

MERGE (o)-[:DELIVERED\_TO]->(city)

MERGE (o)-[:LOCATED\_IN]->(r)

MERGE (o)-[:HAS\_GENDER]->(g)

MERGE (c)-[:HAS\_GENDER]->(gen);

MATCH (o:OrderID)-[r]->(n)

RETURN o, r, n

MATCH (o:OrderID)-[:HAS\_CATEGORY]->(cat:Category)

RETURN cat.name AS Category, SUM(toFloat(o.Profit)) AS TotalProfit

ORDER BY TotalProfit DESC;

MATCH (o:OrderID)-[:HAS\_SUBCATEGORY]->(subcat:SubCategory)

RETURN subcat.name AS SubCategory, SUM(toFloat(o.Profit)) AS TotalProfit

ORDER BY TotalProfit DESC;

MATCH (o:OrderID)-[:HAS\_GENDER]->(g:Gender)

WITH g, o.Day AS day, COUNT(o) AS order\_count

ORDER BY order\_count DESC

WITH g, day, order\_count

WHERE g.name IS NOT NULL

WITH g, COLLECT({day: day, order\_count: order\_count}) AS days

UNWIND days AS day\_info

WITH g, day\_info.day AS day, day\_info.order\_count AS order\_count

ORDER BY g.name, order\_count DESC

WITH g, COLLECT(day) AS top\_days

LIMIT 2

UNWIND top\_days AS top\_day

MERGE (d:Day {name: top\_day})

MERGE (g)-[:PREFERS]->(d)

RETURN g, d

MATCH (o:OrderID)-[:HAS\_CATEGORY]->(cat:Category)

RETURN cat.name AS Category, SUM(toFloat(o.Sales)) AS TotalSales

ORDER BY TotalSales DESC;

MATCH (o:OrderID)-[:HAS\_SUBCATEGORY]->(subcat:SubCategory)

RETURN subcat.name AS SubCategory, SUM(toFloat(o.Sales)) AS TotalSales

ORDER BY TotalSales DESC;

MATCH (o:OrderID)

WITH o.CustomerName AS Customer, SUM(toFloat(o.Sales)) AS TotalSales, SUM(toFloat(o.Profit)) AS TotalProfit

RETURN Customer, TotalSales, TotalProfit

ORDER BY TotalSales DESC;

MATCH (o:OrderID)-[:HAS\_SUBCATEGORY]->(subcat:SubCategory)

MATCH (o)-[:HAS\_GENDER]->(g:Gender)

WHERE g.name = 'Female'

WITH subcat, SUM(toFloat(o.Profit)) AS TotalProfit

ORDER BY TotalProfit DESC

LIMIT 5

RETURN subcat.name AS SubCategory, TotalProfit

ORDER BY TotalProfit DESC;

MATCH (o:OrderID)-[:HAS\_SUBCATEGORY]->(subcat:SubCategory)

MATCH (o)-[:HAS\_GENDER]->(g:Gender)

WHERE g.name = 'Male'

WITH subcat, SUM(toFloat(o.Profit)) AS TotalProfit

ORDER BY TotalProfit DESC

LIMIT 5

RETURN subcat.name AS SubCategory, TotalProfit

ORDER BY TotalProfit DESC;

MATCH (o:OrderID)-[:HAS\_CATEGORY]->(cat:Category)

MATCH (o)-[:LOCATED\_IN]->(r:Region)

WITH r, cat, SUM(toFloat(o.Profit)) AS TotalProfit

ORDER BY TotalProfit DESC

WITH r, collect({category: cat, profit: TotalProfit}) AS TopCategories

UNWIND TopCategories[0..5] AS topCat

RETURN r.name AS Region, topCat.category.name AS Category, topCat.profit AS TotalProfit

ORDER BY TotalProfit DESC;

MATCH (o:OrderID)-[:HAS\_CATEGORY]->(cat:Category)

MATCH (o)-[:LOCATED\_IN]->(r:Region)

WITH r, cat, SUM(toFloat(o.Profit)) AS TotalProfit

ORDER BY TotalProfit DESC

WITH r, collect({category: cat, profit: TotalProfit}) AS TopCategories

UNWIND TopCategories[0..5] AS topCat

MERGE (regionNode:RegionNode {name: r.name})

MERGE (categoryNode:CategoryNode {name: topCat.category.name})

MERGE (regionNode)-[:HAS\_HIGHEST\_PROFIT\_CATEGORY]->(categoryNode)

RETURN regionNode AS RegionNode, categoryNode AS CategoryNode, topCat.profit AS TotalProfit

ORDER BY TotalProfit DESC;

MATCH (o:OrderID)-[:HAS\_CATEGORY]->(cat:Category)

MATCH (o)-[:LOCATED\_IN]->(r:Region)

WITH r, cat, SUM(toFloat(o.Profit)) AS TotalProfit

ORDER BY TotalProfit DESC

WITH r, collect({category: cat, profit: TotalProfit}) AS TopCategories

UNWIND TopCategories[0..5] AS topCat

MERGE (regionNode:RegionNode {name: r.name})

MERGE (categoryNode:CategoryNode {name: topCat.category.name})

MERGE (regionNode)-[:HAS\_HIGHEST\_PROFIT\_CATEGORY]->(categoryNode)

RETURN regionNode AS RegionNode, categoryNode AS CategoryNode, topCat.profit AS TotalProfit

ORDER BY r.name, TotalProfit DESC;

MATCH (o:OrderID)-[:LOCATED\_IN]->(r:Region)

MATCH (o)-[:HAS\_GENDER]->(g:Gender)

RETURN r.name AS Region,

g.name AS Gender,

COUNT(o) AS OrderCount

ORDER BY Region, OrderCount DESC;

MATCH (o:OrderID)-[:LOCATED\_IN]->(r:Region)

MATCH (o)-[:HAS\_GENDER]->(g:Gender)

RETURN r.name AS Region,

g.name AS Gender,

SUM(toFloat(o.Profit)) AS TotalProfit

ORDER BY Region, TotalProfit DESC;

MATCH (o:OrderID)

RETURN o.OrderDate AS Date,

SUM(toFloat(o.Sales)) AS TotalSales

ORDER BY TotalSales DESC;

MATCH (o:OrderID)

RETURN o.OrderDate AS Date,

SUM(toFloat(o.Profit)) AS TotalProfit

ORDER BY TotalProfit DESC;

MATCH (o:OrderID)

WITH o.Day AS DayOfWeek, SUM(toFloat(o.Sales)) AS TotalSales

RETURN DayOfWeek, TotalSales

ORDER BY TotalSales DESC;

MATCH (o:OrderID)

WITH o.Day AS DayOfWeek, SUM(toFloat(o.Profit)) AS TotalProfit

RETURN DayOfWeek, TotalProfit

ORDER BY TotalProfit DESC;

MATCH (o:OrderID)-[:HAS\_GENDER]->(g:Gender)

WITH g, o.Day AS day, COUNT(o) AS order\_count

ORDER BY order\_count DESC

WITH g, day, order\_count

WHERE g.name IS NOT NULL

WITH g, COLLECT({day: day, order\_count: order\_count}) AS days

UNWIND days AS day\_info

WITH g, day\_info.day AS day, day\_info.order\_count AS order\_count

ORDER BY g.name, order\_count DESC

WITH g, COLLECT(day) AS top\_days

LIMIT 3

RETURN g.name AS Gender, top\_days AS Top\_3\_Days

MATCH (o:OrderID)-[:HAS\_GENDER]->(g:Gender), (o)-[:HAS\_CATEGORY]->(cat:Category)

RETURN o.name AS OrderID, g.name AS Gender, cat.name AS Category

LIMIT 10

MATCH (g:Gender), (cat:Category)

RETURN g.name AS Gender, cat.name AS Category

LIMIT 10

MATCH (o:OrderID)-[:HAS\_GENDER]->(g:Gender)

WITH g, o.Day AS day, COUNT(o) AS order\_count

ORDER BY order\_count DESC

WITH g, day, order\_count

WHERE g.name IS NOT NULL

WITH g, COLLECT({day: day, order\_count: order\_count}) AS days

UNWIND days AS day\_info

WITH g, day\_info.day AS day, day\_info.order\_count AS order\_count

ORDER BY g.name, order\_count DESC

WITH g, COLLECT(day) AS top\_days

LIMIT 3

RETURN g.name AS Gender, top\_days AS Top\_3\_Days

MATCH (o:OrderID)-[:HAS\_GENDER]->(g:Gender)

WITH g, o.Day AS day, COUNT(o) AS order\_count

ORDER BY order\_count DESC

WITH g, day, order\_count

WHERE g.name IS NOT NULL

WITH g, COLLECT({day: day, order\_count: order\_count}) AS days

UNWIND days AS day\_info

WITH g, day\_info.day AS day, day\_info.order\_count AS order\_count

ORDER BY g.name, order\_count DESC

WITH g, COLLECT(day) AS top\_days

LIMIT 2

UNWIND top\_days AS top\_day

MERGE (d:Day {name: top\_day})

MERGE (g)-[:PREFERS]->(d)

RETURN g, d

MATCH (o:OrderID)-[:HAS\_GENDER]->(g:Gender),

(o)-[:HAS\_SUBCATEGORY]->(subcat:SubCategory),

(o)-[:LOCATED\_IN]->(r:Region)

WHERE r.name = 'Central' AND g.name IN ['Female', 'Male']

WITH g, subcat, COUNT(o) AS order\_count

WHERE order\_count > 0

WITH subcat, COLLECT(DISTINCT g.name) AS genders, SUM(order\_count) AS total\_orders

WHERE SIZE(genders) = 2  // Kadın ve erkeklerin ortak olarak tercih ettiği alt kategoriler

MATCH (s:SubCategory {name: subcat.name})

MATCH (f:Gender {name: 'Female'}), (m:Gender {name: 'Male'})

MERGE (f)-[:PREFERS]->(s)  // Kadınların tercih ettiği ilişki

MERGE (m)-[:PREFERS]->(s)  // Erkeklerin tercih ettiği ilişki

RETURN f, m, s AS SubCategory, total\_orders

ORDER BY total\_orders DESC

LIMIT 4

MATCH (o:OrderID)-[:HAS\_GENDER]->(g:Gender),

(o)-[:HAS\_SUBCATEGORY]->(subcat:SubCategory),

(o)-[:LOCATED\_IN]->(r:Region)

WHERE r.name = 'West' AND g.name IN ['Female', 'Male']

WITH g, subcat, COUNT(o) AS order\_count

WHERE order\_count > 0

WITH subcat, COLLECT(DISTINCT g.name) AS genders, SUM(order\_count) AS total\_orders

WHERE SIZE(genders) = 2  // Kadın ve erkeklerin ortak olarak tercih ettiği alt kategoriler

MATCH (s:SubCategory {name: subcat.name})

MATCH (f:Gender {name: 'Female'}), (m:Gender {name: 'Male'})

MERGE (f)-[:PREFERS]->(s)  // Kadınların tercih ettiği ilişki

MERGE (m)-[:PREFERS]->(s)  // Erkeklerin tercih ettiği ilişki

RETURN f, m, s AS SubCategory, total\_orders

ORDER BY total\_orders DESC

LIMIT 4

MATCH (o:OrderID)-[:HAS\_GENDER]->(g:Gender),

(o)-[:HAS\_SUBCATEGORY]->(subcat:SubCategory),

(o)-[:LOCATED\_IN]->(r:Region)

WHERE r.name = 'East' AND g.name IN ['Female', 'Male']

WITH g, subcat, COUNT(o) AS order\_count

WHERE order\_count > 0

WITH subcat, COLLECT(DISTINCT g.name) AS genders, SUM(order\_count) AS total\_orders

WHERE SIZE(genders) = 2  // Kadın ve erkeklerin ortak olarak tercih ettiği alt kategoriler

MATCH (s:SubCategory {name: subcat.name})

MATCH (f:Gender {name: 'Female'}), (m:Gender {name: 'Male'})

MERGE (f)-[:PREFERS]->(s)  // Kadınların tercih ettiği ilişki

MERGE (m)-[:PREFERS]->(s)  // Erkeklerin tercih ettiği ilişki

RETURN f, m, s AS SubCategory, total\_orders

ORDER BY total\_orders DESC

LIMIT 4

MATCH (o:OrderID)-[:HAS\_GENDER]->(g:Gender),

(o)-[:HAS\_SUBCATEGORY]->(subcat:SubCategory),

(o)-[:LOCATED\_IN]->(r:Region)

WHERE r.name = 'North' AND g.name IN ['Female', 'Male']

WITH g, subcat, COUNT(o) AS order\_count

WHERE order\_count > 0

WITH subcat, COLLECT(DISTINCT g.name) AS genders, SUM(order\_count) AS total\_orders

WHERE SIZE(genders) = 2  // Kadın ve erkeklerin ortak olarak tercih ettiği alt kategoriler

MATCH (s:SubCategory {name: subcat.name})

MATCH (f:Gender {name: 'Female'}), (m:Gender {name: 'Male'})

MERGE (f)-[:PREFERS]->(s)  // Kadınların tercih ettiği ilişki

MERGE (m)-[:PREFERS]->(s)  // Erkeklerin tercih ettiği ilişki

RETURN f, m, s AS SubCategory, total\_orders

ORDER BY total\_orders DESC

LIMIT 4

MATCH (o:OrderID)-[:HAS\_GENDER]->(g:Gender),

(o)-[:HAS\_SUBCATEGORY]->(subcat:SubCategory),

(o)-[:LOCATED\_IN]->(r:Region)

WHERE r.name = 'South' AND g.name IN ['Female', 'Male']

WITH g, subcat, COUNT(o) AS order\_count

WHERE order\_count > 0

WITH subcat, COLLECT(DISTINCT g.name) AS genders, SUM(order\_count) AS total\_orders

WHERE SIZE(genders) = 2  // Kadın ve erkeklerin ortak olarak tercih ettiği alt kategoriler

MATCH (s:SubCategory {name: subcat.name})

MATCH (f:Gender {name: 'Female'}), (m:Gender {name: 'Male'})

MERGE (f)-[:PREFERS]->(s)  // Kadınların tercih ettiği ilişki

MERGE (m)-[:PREFERS]->(s)  // Erkeklerin tercih ettiği ilişki

RETURN f, m, s AS SubCategory, total\_orders

ORDER BY total\_orders DESC

LIMIT 4

MATCH (o:OrderID)-[:HAS\_GENDER]->(g:Gender)

WHERE g.name IN ['Female', 'Male']

AND o.OrderDate IS NOT NULL

WITH g, toInteger(SUBSTRING(o.OrderDate, 6, 4)) AS year,

SUM(toFloat(o.Profit)) AS total\_profit

WITH year, g.name AS Gender, total\_profit

ORDER BY year, total\_profit DESC

WITH year, COLLECT({gender: Gender, profit: total\_profit}) AS genders

WITH year, genders[0] AS top\_gender

MERGE (y:Year {name: year})  // Yıl node'u oluşturuyoruz

MERGE (g:Gender {name: top\_gender.gender})  // Cinsiyet node'u ile ilişki kuruyoruz

MERGE (y)-[:TOP\_GENDER]->(g)  // Yıl ve cinsiyet arasındaki ilişkiyi ekliyoruz

RETURN y, g

ORDER BY year