### SQL PROJE :Brazilian E-Commerce Public Dataset by Olist

#### **Case 1 : Sipariş Analizi**

##### Question 1 :

-Aylık olarak order dağılımını inceleyiniz. Tarih verisi için order\_approved\_at kullanılmalıdır.

SELECT TO\_CHAR(order\_approved\_at, 'YYYY/MM') AS year\_month,

COUNT(\*) AS order\_count

FROM orders

WHERE order\_approved\_at IS NOT NULL

GROUP BY year\_month

ORDER BY year\_month;

**çizgi, ekran görüntüsü, metin, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Grafikte de görüldüğü üzere ilk bahar ve yaz aylarında önceki aylara nazaran bir sipariş artışı gözlemlenmektedir. Demek ki ilk bahar ve yaz aylarında hareketliliği artırmak adına kampanya çıkılabilir. Ve yine 2017 yaz aylarındaki satış ile 2018 yaz aylarındaki satış miktarı artmış görünmektedir. Bu konu üzerinde belli ki bir çalışma yapılmış. Kış aylarında da özellikle 11.aydaki dikkat çeken artış kampanyaların yoğunlaşmasından kaynaklı olduğu varsayımında bulunabiliriz.**

##### Question 2 :

-Aylık olarak order status kırılımında order sayılarını inceleyiniz. Sorgu sonucunda çıkan outputu excel ile görselleştiriniz. Dramatik bir düşüşün ya da yükselişin olduğu aylar var mı? Veriyi inceleyerek yorumlayınız.

SELECT

TO\_CHAR(order\_approved\_at, 'YYYY/MM') AS monthly, order\_status,

COUNT(\*) AS order\_count

FROM orders

WHERE order\_approved\_at IS NOT NULL

GROUP BY monthly, order\_status

ORDER BY monthly, order\_status;

Sütunlarda DELİVERED ve CANSELED verileri alınmaktadır.

Dramatik bir düşüş yoktur. Aksine teslim edilen verilerde görüldüğü üzere düzenli ve istikrarlı bir artış söz konusudur. 2017/11 verilere göre en çok satış yapılan ay olmuştur. Sebebinin Kasım indirimlerinden kaynaklı olduğunu düşünmekteyim.

##### Question 3 :

-Ürün kategorisi kırılımında sipariş sayılarını inceleyiniz. Özel günlerde öne çıkan kategoriler nelerdir? Örneğin yılbaşı, sevgililer günü…

SELECT

product\_category\_name,

COUNT(\*) AS order\_count

FROM orders o

JOIN order\_items oi ON o.order\_id = oi.order\_id

JOIN products p ON oi.product\_id = p.product\_id

WHERE

(DATE(o.order\_approved\_at) <= '2018-02-14' and DATE(o.order\_approved\_at) > '2018-02-01')

or (DATE(o.order\_approved\_at) <= '2015-02-14' and DATE(o.order\_approved\_at) > '2015-02-01')

or (DATE(o.order\_approved\_at) <= '2016-02-14' and DATE(o.order\_approved\_at) > '2016-02-01')

or (DATE(o.order\_approved\_at) <= '2017-02-14' and DATE(o.order\_approved\_at) > '2017-02-01')

GROUP BY product\_category\_name

ORDER BY order\_count DESC;

**3 yıllık şubat 1-14 arası sipariş tarihi incelendi. (14 Şubat için bu tarih aralığı incelendi.)**

Sevgililer günü özelinde 14 günlük filtreleme sonucunda en çok satış yapılan alan ev ürünleri olan ‘yatak-masa-banyo’ kategorisi olmuştur. Sonrasındaki ‘bilişim aksesuarları’ kategorisi olmuştur. Sırayla ‘güzellik, spor-eğlence, dekorasyon-mobilya’ kategorisi olmuştur.

##### Question 4 :

-Haftanın günleri(pazartesi, perşembe, ….) ve ay günleri (ayın 1’i,2’si gibi) bazında order sayılarını inceleyiniz. Yazdığınız sorgunun outputu ile excel’de bir görsel oluşturup yorumlayınız.

**HAFTANIN GÜNLERİ**

**SELECT**

**TO\_CHAR(order\_approved\_at, 'Day') AS days,**

**COUNT(\*) AS order\_count**

**FROM orders**

**WHERE order\_approved\_at IS NOT NULL**

**GROUP BY TO\_CHAR(order\_approved\_at, 'Day')**

**ORDER BY MIN(EXTRACT(DOW FROM order\_approved\_at));**

**Araştırmalarım sonucu hafta günleri için postgresql de DOW fonksiyonunu kullandım.**

**Grafikte de görüldüğü üzere en düşük sipariş sayısı Pazar günü alınmış. En yüksek sipariş sayısı ise Salı günü yapılmış görünmektedir. Akabinde Çarşamba, Perşembe, Cuma, Pazartesi ve Cumartesi günleri sayısına göre sıralanmaktadır. Şöyle bir varsayımda bulunabiliriz, insanlar hafta sonu kendilerine daha fazla vakit ayırmak istedikleri için alışverişe minimum vakit ayırıyor olabilirler. Ayrıca hafta içi iş yoğunluğu ve günlük stresini atmak için alışveriş yapma oranları artıyor olabilir. Bununla alakalı en çok alışveriş yapılan kategoriler üzerinden bir kampanya çıkılabilir veyahut sipariş sayısı az olan ürünler ile alakalı hareketliliği artırmak adına indirim, kampanya yapılabilir.**

**AYIN GÜNLERİ**

**SELECT**

**EXTRACT(DAY FROM order\_approved\_at) AS days,**

**COUNT(\*) AS order\_count**

**FROM orders**

**WHERE order\_approved\_at IS NOT NULL**

**GROUP BY days**

**ORDER BY days;**

**Ayın son günlerine doğru sipariş sayılarında bir düşüş gözlemlenmekte, bunu da maaş günlerinin genellikle ayın başı ya da ayın ortalarında olması ile ilişkilendirdim. Ortalama maaş günleri bazlı bir planlama yapılıp satış hareketliliği sağlanabilir.**

#### **Case 2 : Müşteri Analizi**

##### Question 1 :

-Hangi şehirlerdeki müşteriler daha çok alışveriş yapıyor? Müşterinin şehrini en çok sipariş verdiği şehir olarak belirleyip analizi ona göre yapınız.

**Örneğin; Sibel Çanakkale’den 3, Muğla’dan 8 ve İstanbul’dan 10 sipariş olmak üzere 3 farklı şehirden sipariş veriyor. Sibel’in şehrini en çok sipariş verdiği şehir olan İstanbul olarak seçmelisiniz ve Sibel’in yaptığı siparişleri İstanbul’dan 21 sipariş vermiş şekilde görünmelidir.**

**SELECT customer\_city,**

**COUNT(order\_id) AS order\_count**

**FROM (**

**SELECT c.customer\_city,**

**o.order\_id**

**FROM orders o**

**LEFT JOIN customers c ON c.customer\_id = o.customer\_id**

**) AS subquery**

**GROUP BY customer\_city**

**ORDER BY order\_count DESC**

**limit 10;**

En çok satış yapılan 10 şehri filtreledim ve yukarıdaki grafiği elde ettim. Bu şehirler üzerinden hangi kategoride daha çok satış yapılıyor veya ortalama satış yapılan ürünler hangileri bakılabilir. Ayrıca hangi müşterilerin aktif olarak alışveriş yaptığı tespit edilip onlara yönelik ek indirimler yapılabilir. Aynı zamanda orta segment müşterilerde de aynı yol izlenebilir.

#### **Case 3: Satıcı Analizi**

##### Question 1 :

-Siparişleri en hızlı şekilde müşterilere ulaştıran satıcılar kimlerdir? Top 5 getiriniz. Bu satıcıların order sayıları ile ürünlerindeki yorumlar ve puanlamaları inceleyiniz ve yorumlayınız.

İlk 5 satıcıyı getiren sorgu:

SELECT

s.seller\_id,

AVG(order\_delivered\_customer\_date - order\_purchase\_timestamp) AS avg\_delivery\_time

FROM orders o

JOIN order\_items oi ON o.order\_id = oi.order\_id

JOIN sellers s ON oi.seller\_id = s.seller\_id

WHERE o.order\_delivered\_customer\_date IS NOT NULL

GROUP BY s.seller\_id

ORDER BY avg\_delivery\_time ASC

LIMIT 5;

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

Sorguda en hızlı teslimatı bulmak için order\_delivered\_customer\_date verisinden order\_purchase\_timestamp verisini çıkararak teslimat süresine ulaştım. Siparişin verilme ve teslim edilme süreci ortalama 1 gün ila 2 gün arasında yapılmaktadır. En hızlı satıcı 1 gün 5 saatlik bir süreçte teslimatı müşteriye ulaştırmıştır.

**Yorum ve puanlamayı içeren sorgu:**

WITH fast\_sellers AS (

SELECT

s.seller\_id,

AVG(order\_delivered\_customer\_date - order\_purchase\_timestamp) AS avg\_delivery\_time

FROM orders o

JOIN order\_items oi ON o.order\_id = oi.order\_id

JOIN sellers s ON oi.seller\_id = s.seller\_id

WHERE o.order\_delivered\_customer\_date IS NOT NULL

GROUP BY s.seller\_id

ORDER BY avg\_delivery\_time ASC

LIMIT 5

)

SELECT fs.seller\_id,

COUNT(DISTINCT oi.order\_id) AS order\_count,

ROUND(AVG(r.review\_score)) AS avg\_review\_score,

COUNT(DISTINCT r.review\_comment\_message) AS total\_review\_message

FROM fast\_sellers fs

LEFT JOIN order\_items oi ON oi.seller\_id = fs.seller\_id

LEFT JOIN reviews AS r ON r.order\_id = oi.order\_id

GROUP BY fs.seller\_id;

Yorumlama ile hızlı teslimat arasında anlamlı bir ilişki kurulamamıştır. Hızlı teslimat yapan satıcıların puanlarının yüksek olduğu gözlemlenmektedir.

metin, çizgi, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

##### Question 2 :

-Hangi satıcılar daha fazla kategoriye ait ürün satışı yapmaktadır?

Fazla kategoriye sahip satıcıların order sayıları da fazla mı?

WITH seller\_category\_orders AS (

SELECT

s.seller\_id,

COUNT(DISTINCT p.product\_category\_name) AS count\_categories,

COUNT(o.order\_id) AS count\_orders

FROM

orders o

JOIN order\_items oi ON o.order\_id = oi.order\_id

JOIN products p ON oi.product\_id = p.product\_id

JOIN sellers s ON oi.seller\_id = s.seller\_id

GROUP BY

s.seller\_id

)

SELECT

s.seller\_id,

s.count\_categories,

s.count\_orders

FROM

seller\_category\_orders s

ORDER BY

s.count\_categories DESC, s.count\_orders DESC;

Aşağıdaki görsellerde fazla kategoriye sahip satıcıların sipariş sayıları ve kategori sayıları vardır. Kategori fazlalığı ile sipariş sayısının fazla olması arasında doğru orantılı bir ilişki kurulmuştur. Ancak fazla kategoriye kıyasla daha az kategoriye sahip satıcıların sipariş sayılarının daha fazla olduğu durumlarda oldukça fazladır.

metin, sayı, numara, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

#### **Case 4 : Payment Analizi**

##### Question 1 :

-Ödeme yaparken taksit sayısı fazla olan kullanıcılar en çok hangi bölgede yaşamaktadır? Bu çıktıyı yorumlayınız.

SELECT

p.payment\_installments,

c.customer\_city,

COUNT(DISTINCT o.customer\_id) AS customer\_count

FROM payments AS p

LEFT JOIN orders AS o ON p.order\_id = o.order\_id

LEFT JOIN customers AS c ON o.customer\_id = c.customer\_id

GROUP BY 1,

2

ORDER BY 1 desc,

3 desc;

20-24 taksit arası yapılan taksitlerin alıcılarının bulunduğu bölgeler alınmıştır. İstenirse daha az taksitli alışveriş yapılan bölgelere de ulaşılabilir. En fazla olan istendiği için 20-24 arası yapılan taksitlerin bölgeleri filtrelenmiştir.

Taksit sayısının fazla olması ile sipariş sayısının artması ile alakalı anlamlı bir ilişki kurulamamıştır. Görüldüğü üzere sipariş sayıları oldukça azdır. Şöyle bir durumda söz konusu olabilir pahada ağır ürünler de fazla taksit yoluna gidiyor olabilir müşteriler ve bundan ötürü sipariş sayısı az olabilir lakin yine de taksit sayısı düşürülebilir ya da biz bu bölgeleri kazanmak istiyorsak bölge bazlı hareketliliği artıracak farklı indirimler tanımlanabilir.

##### Question 2 :

-Ödeme tipine göre başarılı order sayısı ve toplam başarılı ödeme tutarını hesaplayınız. En çok kullanılan ödeme tipinden en az olana göre sıralayınız.

**SELECT payment\_type,**

**COUNT(DISTINCT o.order\_id) AS successful\_order\_count,**

**TO\_CHAR(SUM(payment\_value),'FM999,999,999.00') AS total\_successful\_payment\_amount**

**FROM orders o**

**JOIN payments p ON o.order\_id = p.order\_id**

**WHERE order\_status = 'delivered'**

**GROUP BY payment\_type**

**ORDER BY successful\_order\_count DESC;**

**metin, ekran görüntüsü, yazı tipi, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu**

##### Question 3 :

-Tek çekimde ve taksitle ödenen siparişlerin kategori bazlı analizini yapınız. En çok hangi kategorilerde taksitle ödeme kullanılmaktadır?

En çok taksit yapılanlara göre sorgu:

SELECT

p.payment\_installments,

pr.product\_category\_name,

COUNT(DISTINCT o.order\_id) AS order\_count

FROM orders AS o

LEFT JOIN payments AS p ON p.order\_id = o.order\_id

LEFT JOIN order\_items AS oi ON oi.order\_id = o.order\_id

LEFT JOIN products AS pr ON pr.product\_id = oi.product\_id

WHERE p.payment\_installments > 1

AND pr.product\_category\_name IS NOT NULL

GROUP BY p.payment\_installments, pr.product\_category\_name

ORDER BY payment\_installments DESC, order\_count DESC;

En çok taksit yapılan kategoriler aşağıda bir görselle sıralanmıştır.

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Taksitli olarak 500 ve üzeri sipariş alan kategoriler yukarıda görselleştirilmiştir.

Ödeme tipi tek çekim olan siparişlerin sorgusu:

SELECT

p.payment\_installments,

pr.product\_category\_name,

COUNT(DISTINCT o.order\_id) AS order\_count

FROM orders AS o

LEFT JOIN payments AS p ON p.order\_id = o.order\_id

LEFT JOIN order\_items AS oi ON oi.order\_id = o.order\_id

LEFT JOIN products AS pr ON pr.product\_id = oi.product\_id

WHERE p.payment\_installments = 1

AND pr.product\_category\_name IS NOT NULL

GROUP BY p.payment\_installments, pr.product\_category\_name

ORDER BY order\_count DESC, payment\_installments DESC;

metin, sayı, numara, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Ortalama aynı kategorilerde taksit veya tek çekim yapıldığı görülmektedir. Fazla taksitli siparişlerde, sipariş sayısı oldukça azdır.

#### **Case 5 : RFM Analizi**

Aşağıdaki e\_commerce\_data\_.csv doyasındaki veri setini kullanarak RFM analizi yapıldı.   
Recency hesaplarken bugünün tarihi değil en son sipariş tarihi baz alındı.

Veri seti bu linkten alınmıştır, veriyi tanımak için linke girip inceleyebilirsiniz.  
[E-Commerce Data](https://www.kaggle.com/datasets/carrie1/ecommerce-data)  
  
--"2011-12-09"-- Problem yaşamamak için maksimum invoicedate i aldım.

WITH max\_i\_d AS (

SELECT customer\_id,

MAX(invoicedate) AS max\_invoicedate

FROM rfm

WHERE customer\_id IS NOT NULL

GROUP BY customer\_id

ORDER BY 2 DESC

),

recency AS (

SELECT customer\_id,

max\_invoicedate,

('2011-12-09'::date - max\_invoicedate:: date) AS recency

FROM max\_i\_d

WHERE customer\_id IS NOT NULL

),

frequency AS (

SELECT customer\_id,

COUNT( DISTINCT customer\_id) AS frequency

FROM rfm

WHERE customer\_id IS NOT NULL

GROUP BY customer\_id

),

monetary AS(

SELECT customer\_id,

ROUND(SUM(unitprice)::numeric, 0)AS monetary

FROM rfm

WHERE customer\_id IS NOT NULL

GROUP BY customer\_id

),

scores AS(

SELECT

r.customer\_id,

r.recency,

NTILE (5) OVER(ORDER BY recency DESC) AS recency\_score,

f.frequency,

CASE WHEN f.frequency >=1 and f.frequency<=4 THEN f.frequency ELSE 5 end AS frequency\_score,

m.monetary,

NTILE (5) OVER(ORDER BY monetary asc) AS monetary\_score

FROM recency r

LEFT JOIN frequency f on r.customer\_id= f.customer\_id

LEFT JOIN monetary m on f.customer\_id= m.customer\_id

),

montery\_frequency AS (

SELECT customer\_id,

recency\_score,

frequency\_score + monetary\_score AS mon\_fre\_score

FROM scores

),

rfm\_score as(

SELECT customer\_id,

recency\_score,

NTILE(5) OVER(ORDER BY mon\_fre\_score) AS mon\_fre\_score

FROM montery\_frequency

)

SELECT\*FROM rfm\_score