

## Database Bootcamp – Ödev 2

Bu ödevde, verilen SQL sorularına uygun sorgular yazıldı ve her birinin açıklaması detaylandırıldı.

**1.Soru: Invoice tablosunda, tüm değerleri NULL olan kayıtların sayısını bulmanız isteniyor.**

**Bu işlemi tek bir sorgu ile yapmalısınız.**

**Sorguyu yazdıktan sonra, PostgreSQL'in sol alt kısmındaki Row sayısını, SQL sorgunuzda yorum satırında belirtmeniz gerekmektedir.**

```
1  ✓ SELECT * FROM invoice
2  WHERE invoice_id IS NULL
3         AND customer_id IS NULL
4         AND invoice_date IS NULL
5         AND billing_address IS NULL
6         AND billing_city IS NULL
7         AND billing_state IS NULL
8         AND billing_country IS NULL
9         AND billingpostal_code IS NULL
10        AND total IS NULL;
11
12  --total row = 0
```

Bu sorguda invoice tablosundaki tüm sütunları seçerek başladım.

Daha sonra, sütunlardaki **NULL** değerleri kontrol etmek için **IS NULL** operatörünü kullandım. **AND** operatörü ise tüm sütunların aynı anda **NULL** olup olmadığını kontrol ederek, yalnızca bu koşulu sağlayan satırları seçmemi sağlıyor.

**AND** operatörü yerine **OR** operatörü kullanılsaydı örnek olarak billing\_state'i NULL olan ama invoice\_date NULL olmayan satırlarda dönerdi.

SQL' de yorum satırı tek bir satır olacaksa – şekilde yapılabilir. Daha uzun yorum satırları için /\*...../\* ifadesi kullanılır.

**2.Soru: Koordinasyondaki kişiler, Total değerlerinde bir hata olduğunu belirtiyorlar. Bu değerlerin iki katını görmek ve eski versiyonlarıyla birlikte karşılaştırmak için bir sorgu yazmanız isteniyor.**

Ayrıca, verilerin daha rahat takip edilebilmesi için, tablonun yeni versiyonuna ait kolona göre büyükten küçüğe sıralama yapılması isteniyor.

```
1 SELECT total, 2 * total AS new_total FROM invoice
2 ORDER BY new_total DESC
```

Data Output Messages Notifications

	total numeric (10,2) 🔒	new_total numeric 🔒
1	25.86	51.72
2	23.86	47.72
3	21.86	43.72
4	21.86	43.72
5	18.86	37.72
6	18.86	37.72

Bu sorguda, invoice tablosundaki total sütununu 2 ile çarptım ve **AS** operatörü ile sonucu new\_total adıyla bir sütun olarak adlandırdım. Daha sonra, **ORDER BY** ifadesiyle new\_total sütununu büyükten küçüğe sıralamak için **DESC** operatörünü kullandım.

**3.Soru: Adres kolonundaki verileri, soldan 3 karakter ve sağdan 4 karakter olarak birleştirmeniz ve "Açık Adres" olarak yazmanız isteniyor.**

**Ayrıca, bu yeni açık adresi 2013 yılı ve 8. ay'a göre filtrelemeniz gerekiyor.**

1	SELECT (SUBSTRING(billing_address, 1, 3))
2	(SUBSTRING(billing_address, LENGTH(billing_address) - 3, 4))
3	AS acik_adres, invoice_date
4	FROM invoice
5	
6	WHERE EXTRACT(YEAR FROM invoice_date) = 2013
7	AND EXTRACT(MONTH FROM invoice_date) = 8;
8	

  

Data Output	Messages	Notifications
-------------	----------	---------------

  

+	SQL
---	-----

  

	acik_adres text	invoice_date timestamp without time zone
1	3 Creet	2013-08-02 00:00:00
2	Lij20bg	2013-08-02 00:00:00
3	C/ o 85	2013-08-03 00:00:00
4	110n Pl	2013-08-04 00:00:00
5	Av.2170	2013-08-07 00:00:00
6	Rua 155	2013-08-12 00:00:00
7	162reet	2013-08-20 00:00:00

Bu sorguda, invoice tablosundaki billing\_address ve invoice\_date sütunları SELECT ifadesi ile seçilmiştir.

- **SUBSTRING** () fonksiyonu kullanılarak, billing\_address değerinin ilk 3 karakteri ve son 4 karakteri alınmış, || operatörü ile birleştirilmiş ve AS ifadesiyle acik\_adres olarak adlandırılmıştır.
- **SUBSTRING** (billing\_address, **LENGTH** (billing\_address)- 3, 4) işleminde, billing\_address değerinin uzunluğundan 3 çıkarılmasının sebebi, **SUBSTRING** fonksiyonunun **1'den başlayarak indeksleme yapmasıdır**. Eğer **SUBSTRING** (billing\_address, **LENGTH** (billing\_address)- 4, 4) şeklinde kullanılsaydı, son 4 karakter yerine **son 3 karakter alınmış olurdu**.

Alternatif olarak, **LEFT()** ve **RIGHT()** fonksiyonlarıyla daha okunaklı bir şekilde aynı işlem yapılabilir:

1	SELECT
2	LEFT(billing_address, 3)    RIGHT(billing_address, 4)
3	AS acik_adres,
4	invoice_date
5	FROM invoice
6	WHERE EXTRACT(YEAR FROM invoice_date) = 2013
7	AND EXTRACT(MONTH FROM invoice_date) = 8;

  

Data Output	Messages	Notifications
-------------	----------	---------------

  

≡	📄	▼	📋	▼	🗑️	📦	⬇️	📈	SQL
---	---	---	---	---	----	---	----	---	-----

  

	acik_adres text	invoice_date timestamp without time zone
1	3 Creet	2013-08-02 00:00:00
2	Lij20bg	2013-08-02 00:00:00
3	C/ o 85	2013-08-03 00:00:00
4	110n Pl	2013-08-04 00:00:00
5	Av.2170	2013-08-07 00:00:00
6	Rua 155	2013-08-12 00:00:00
7	162reet	2013-08-20 00:00:00

Bu yöntemde:

- **LEFT** (billing\_address, 3), adresin ilk 3 karakterini alır.
- **RIGHT** (billing\_address, 4), adresin son 4 karakterini alır.

|| operatörü **PostgreSQL'e özgüdür**. Farklı bir veritabanı yönetim sistemi kullanılıyorsa, sorgunun CONCAT fonksiyonu ile şu şekilde yazılması gerekir:

```
SELECT CONCAT (SUBSTRING (billing_address, 1, 3),
SUBSTRING (billing_address, LENGTH (billing_address)- 3, 4))
AS acik_adres, invoice_date
FROM invoice)
```

Sorgunun WHERE koşulunda, **EXTRACT** (YEAR FROM invoice\_date) = 2013 ifadesiyle yalnızca 2013 yılına ait faturalar seçilmiştir. Ayrıca, **EXTRACT** (MONTH FROM invoice\_date) = 8 ifadesiyle bu faturaların sadece ağustos ayında kesilmiş olanları filtrelenmiştir.