ORACLE SQL RAPORU

Hilal Uluca

ORACLE SQL RAPORU

# GİRİŞ

Projenin amacı SQL temellerini, fonksiyonlarını, join, alt sorgular ve Oracle’a özgü özellikleri uygulamaktır. Kullandığım veri tabanı, Oracle SQL Live içerisindeki Oracle HR şemasıdır.

# VERİ MODELİ

Aşağıda HR (Humon Resources) veri tabanının ilişkilerini göstermek adına diyagram verilmiştir:

HR.DEPARTMENTS

* Department\_id
* Department\_name
* Manager\_id
* Location\_id

HR.EMPLOYEES

* Employee\_id
* First\_name
* Last\_name
* Email
* Phone\_number
* Hire\_date
* Job\_id
* Salary
* Commission\_pct
* Manager\_id
* Department\_id

HR.REGIONS

* Region\_id
* Region\_name

HR.COUNTRIES

* Country\_id
* Country\_name
* Region\_id

HR.JOBS

* Job\_id
* Job\_title
* Min\_salary
* Max\_salary

HR.LOCATIONS

* Location\_id
* Street\_address
* Postal\_code
* City
* State\_province
* Country\_id

HR.JOB\_HISTORY

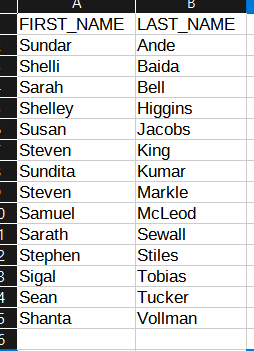
* Employee\_id
* Start\_date
* End\_date
* Job\_id
* Department\_id

# SORGULAR & ÇIKTILAR

1. Bu sorguda select, where ve like kullanılarak ismi S ile başlayan çalışanlar listelenmiştir.

select FIRST\_NAME, LAST\_NAME from hr.EMPLOYEES

where FIRST\_NAME LIKE 'S%';



1. Bu sorguda where, hire\_date, extract kullanılarak 2010 yılında işe alınan çalışanlar listelenmiştir. Herhangi bir çıktıya ulaşılamamıştır.

select FIRST\_NAME, LAST\_NAME, HIRE\_DATE from hr.EMPLOYEES

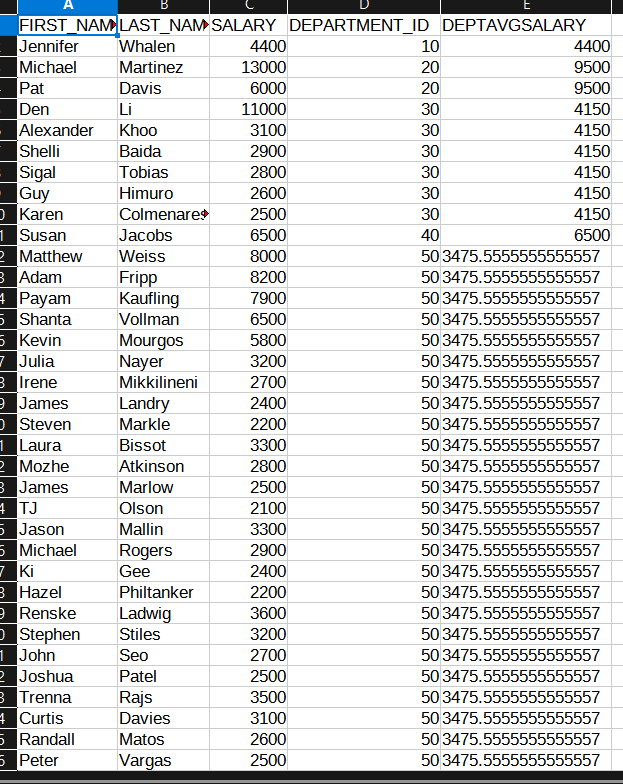
where EXTRACT(year from HIRE\_DATE)=2010;

1. Bu sorguda **analitik fonksiyonlardan**  AVG,OVER(PARTITION BY ...) kullandım.

select first\_name,last\_name,salary,DEPARTMENT\_ID,

avg(salary) over (partition by DEPARTMENT\_ID) as deptavgsalary

from hr.EMPLOYEES;



1. Bu sorguda string fonksiyonlarından concat kullanarak isim soyisim sütununu birleştirildi ve sonuna @company.name eklenmesi sağlandı.

select concat(first\_name,last\_name)|| '@company.com' from hr.employees;



1. Bu sorguda sayısal fonksiyonlardan ceil ve floor kullanarak çalışan maaşlarının tavan ve aşağı yuvarlanması sağlanmıştır.

select first\_name,last\_name, DEPARTMENT\_ID, ceil(salary), floor(salary)

from hr.employees

where DEPARTMENT\_ID in (10,11,12,13,14);

C:\Users\hilal\OneDrive\Desktop\Resimler\Screenshots\Ekran görüntüsü 2025-09-15 164839.png

1. Bu sorguda tarih fonksiyonlarından sysdate kullanarak çalışanların giriş tarihinden itibaren kaç gündür şirkette olduklarının bulunması sağlanmıştır.

select FIRST\_NAME,LAST\_NAME,(sysdate-hire\_date) from hr.employees

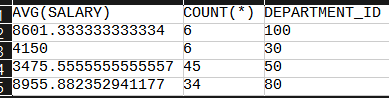


1. Bu sorguda group by, having ve aggregation kullanılarak çalışan sayısı 5’ten fazla olan departmanlar ve ortalama maaşların bulunması sağlanmıştır.

select avg(salary),count(\*), DEPARTMENT\_ID from hr.employees

group by DEPARTMENT\_ID

having count(\*)>5;



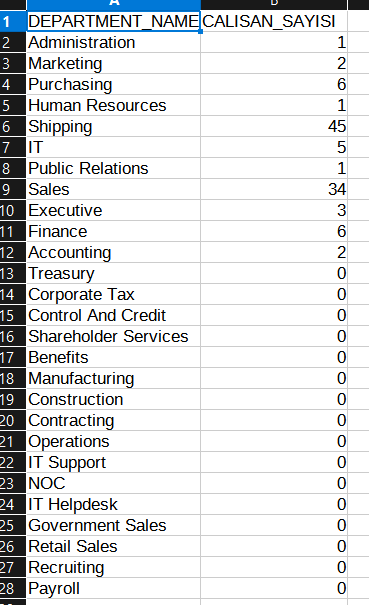
1. Bu sorguda joinlerden right join kullanılarak her departmandaki çalışan sayısı listelenmiştir.

Select d.department\_name, COUNT(e.employee\_id) as calisan\_sayisi from hr.employees e

right join hr.departments d

on e.department\_id = d.department\_id

group by d.department\_name;



1. Bu sorguda alt sorgulardan exists kullanılarak prim alan çalışanlar listelenmiştir.

Select first\_name,last\_name, commission\_pct from hr..employees e

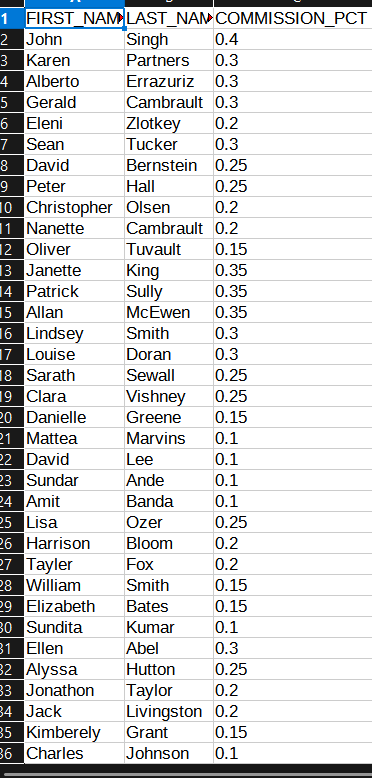
where exists

( select 1 from hr.employees

where commission\_pct is not null

and employee\_id = e.employee\_id

);



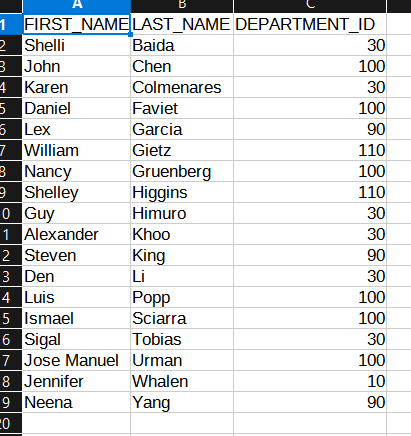
1. Burada alt sorgulardan in kullanılarak lokasyonu 1700 olan departmanlarda çalışanlar listelenmiştir.

select first\_name,last\_name,department\_id from hr.EMPLOYEES

where department\_id in

(select department\_id from hr.departments

where location\_id=1700);



1. Bu sorguda with / recursive cte kullanarak yönetici hiyerarşisini ve kaçıncı seviyede oldukları listelenmiştir.

with yonetici\_hiyer (employee\_id, first\_name, last\_name, manager\_id, seviye) as (

select employee\_id,first\_name,last\_name,manager\_id,1 as seviye from hr.employees

where manager\_id is null

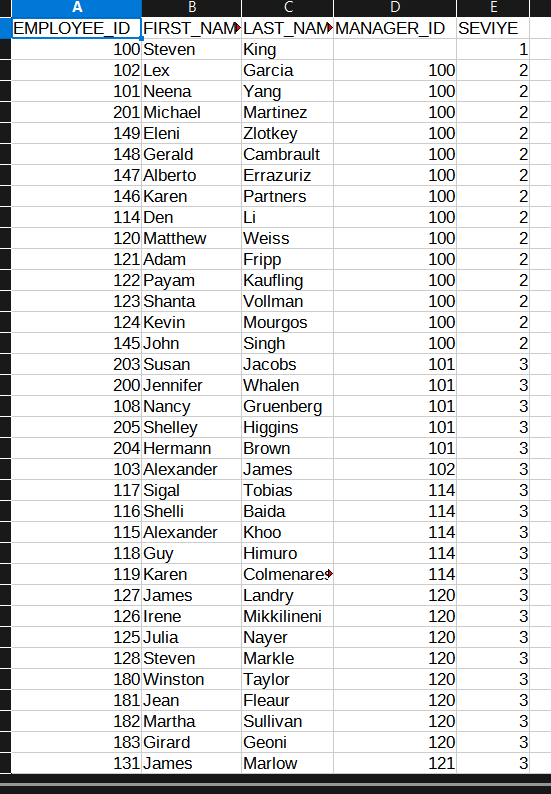
union all

select e.employee\_id,e.first\_name,e.last\_name,e.manager\_id,yh.seviye + 1 from hr.employees e

inner join yonetici\_hiyer yh

on e.manager\_id = yh.employee\_id)

select \*from yonetici\_hiyer order by seviye, manager\_id;



1. Bu sorguda insert kullanılarak yeni bir çalışanın bilgileri eklenmiştir.

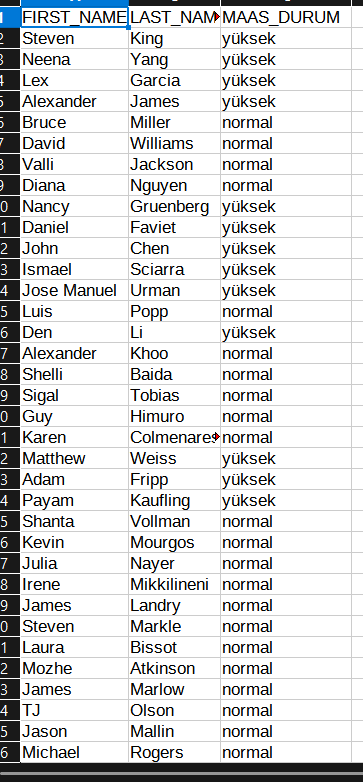
İnsert into hr.employees (employee\_id, first\_name, last\_name, department\_id, salary, hire\_date)

values (employees\_seq.nextval, ‘Ali’, ‘Yılmaz’, 10, 5500, sysdate);

1. Bu sorguda Oracle’a özgü fonksiyonlardan olan decode kullanılmış, maaşı 7000’den yüksek olan çalışanlara ‘yüksek’ değilse ‘normal’ yazılmıştır.

select first\_name, last\_name, decode(sign(salary - 7000), 1, 'yüksek', 'normal') as maas\_durum

from hr.employees;



## 4-SONUÇ

Bu proje kapsamında Oracle HR veri tabanı üzerinde SQL’in temel konuları (seçim, filtreleme, sıralama), fonksiyonlar (string, sayısal, tarih), JOIN işlemleri, alt sorgular, aggregation, analitik fonksiyonlar ve Oracle’a özgü özellikler uygulanmıştır.

Raporda yer alan SQL çıktıları, sadece örnek teşkil etmesi amacıyla sınırlı sayıda gösterilmiş, tüm sorgular Oracle SQL Live ortamında çalıştırılmıştır.  
 Projeyle birlikte hem SQL sorgulama pratiği kazanılmış hem de farklı sorgu tiplerinin gerçek bir veri modeli üzerinde nasıl çalıştığı gözlemlenmiştir.  
Bu çalışma ileride daha gelişmiş veri analizi ve raporlama projeleri için sağlam bir temel oluşturmaktadır.