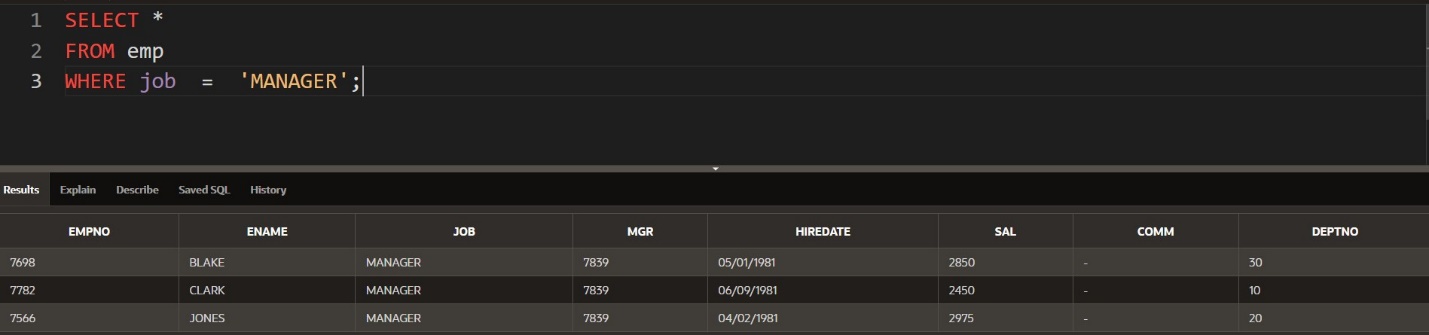
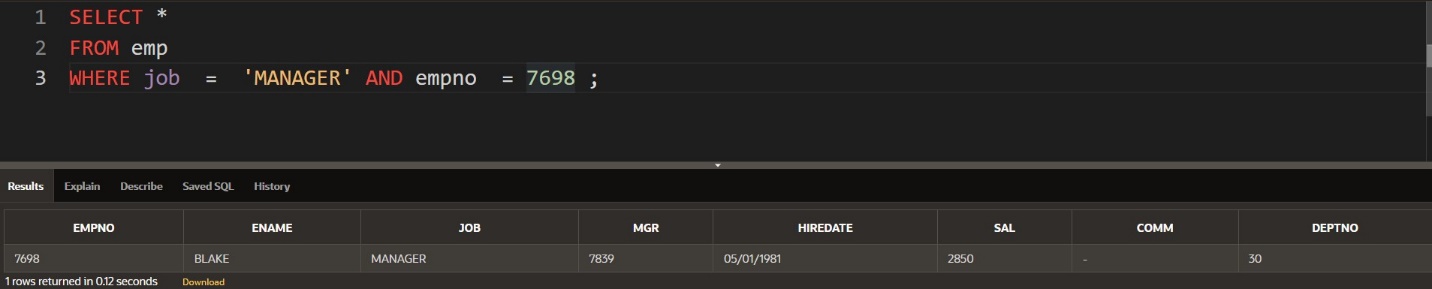
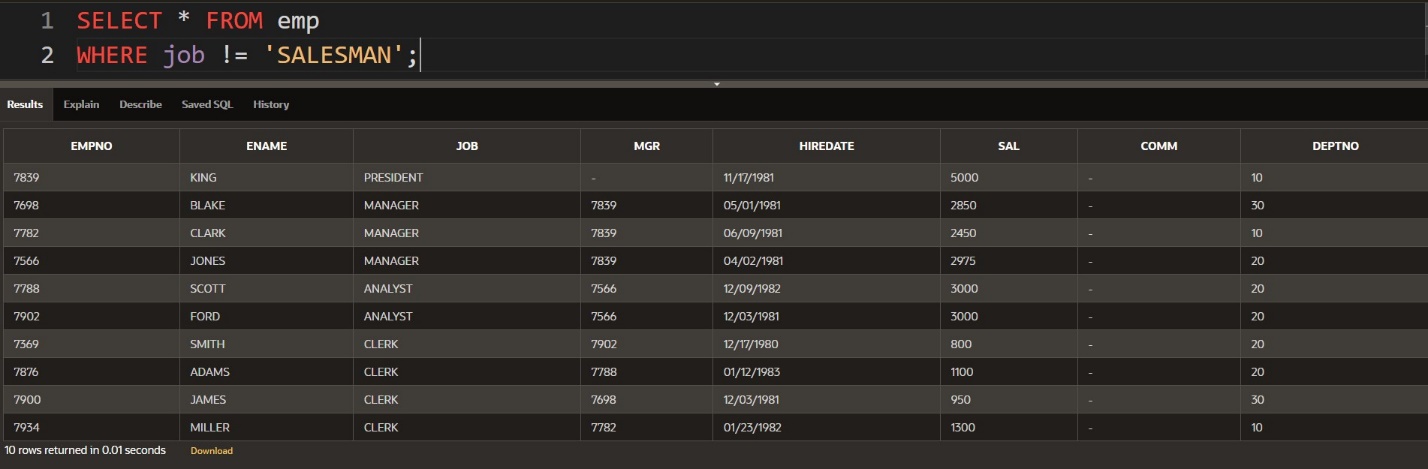
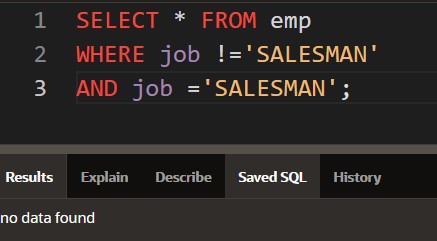
1. Keçən dərslikdə biz **SQL** də **WHERE** clause-nı, yəni, filterlama əməliyyatını öyrəndik və həmçinin birdən çox sütuna görə filterlama əməliyyatını həyata keçirtmək üçün **AND** operatorunun istifadə qaydasını öyrənmiş olduq. Bu dərslikdə isə biz, keçən dərsliyin davamını əlavə operatorlarla istifadə qaydasını öyrənmiş olacayıq. Qısası öyrənəcəyik ki, **WHERE** keyword-ü ilə birlikdə başqa hansı operatorlar istifadə olunur ona nəzər yetirəcəyik.
2. Operatorlar **SQL** də bir sütun ilə digər sütun arasında müqayisə etmə əməliyyatını həyata keçirtmək üçün istifadə olunur.
3. Keçən dərslikdə biz **WHERE** clause-ında birdən çox sütuna görə filterlama əməliyyatını **AND** operatorundan istifadə qaydasını öyrənmiş olduq və həmçinin biz **=(equal)** operatorunuda **WHERE** clause-ında istifadə qaydasını öyrənmiş olduq. Məsələn aşağıdakı şəkildəki kimi.

yuxarıdakı şəkildə gördüyünüz kimi **=(equal)** və **AND** operatorlarının istifadə qaydasını göstərilmişdir.

1. Gəlin elə isə yeni bir operator ilə tanış olaq və anlamağa çalışaq ki, bu necə işləyir? Məsələn fərz edin ki, bizim **emp** table-ından **job** column-da dəyəri **‘SALESMAN’ olmayan** işçilərin məlumatlarını əldə edirik, bu zaman nə edəcəyik, əlbətdəki aşağıdakı şəkildə olan query-ni yazıb çalışdırsaq istədiyimiz nəticəni əldə etmiş olarıq.

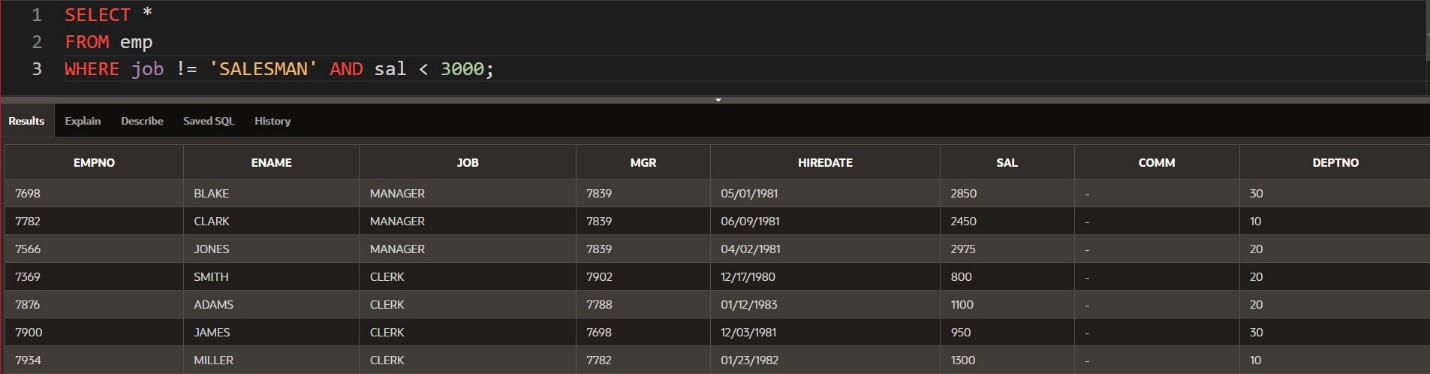
yuxarıdakı şəkildə **job** sütununda dəyəri **‘SALESMAN’** olmayan işçilərin məlumatlarını əldə etdik. Bunu həyata biz, **!= (not equal)** operatoru vasitəsi ilə həyata keçirtmiş olduq. Bu operatorun mənası bu deməkdir ki, bütün dəyərləri ver, **SALESMAN**-ı çıxmaq şərti ilə.

1. **SQL** də **WHERE** clause-ı condition-lar ilə birlikdə **table** da olan hər bir row-a uyğun gələn condition-a görə check əməliyyatın apararaq həyata keçirdir. Fərz eləyin ki, aşağıdakı kimi bizim bir query-miz var və gəlin onu analiz etməyə başlıyaq.

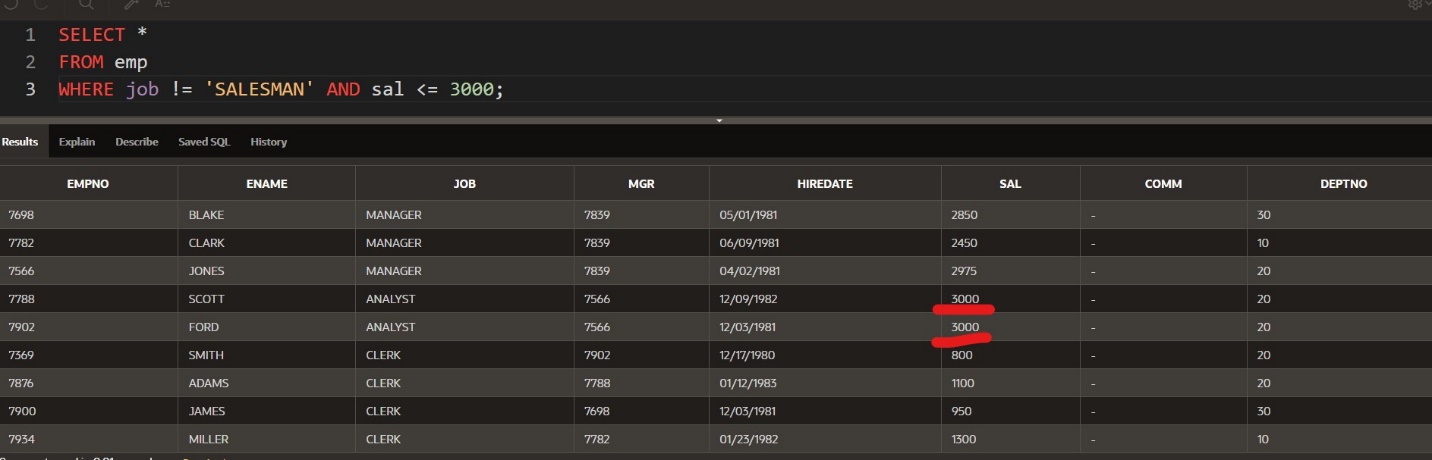


yuxarıdakı şəkildə olan query-ni işə salıb görsək bizə bir boş **resultset** qayıtmış olacaqdır, buna səbəb odur ki, **SQL** az öncədə dediyimiz kimi hər bir row-a görə condition controller əməliyyatını həyata keçirtdiyindən burada fərz eləyin ki, məsələn girdiyi ilk **row** da apardığı check controller əməliyyatı condition ilə tutuşa bilər yəni record qaytara bilər, lakin **AND** tərəfində də həmçinin başqa bir şərt yazıldığından və həmən record üzərində bu check controller əməliyyatını apardığından condition tutuşmaya bilər və belə olduğu halda bizə heç bir dolu **resultset** qayıtmıyacaqdır.

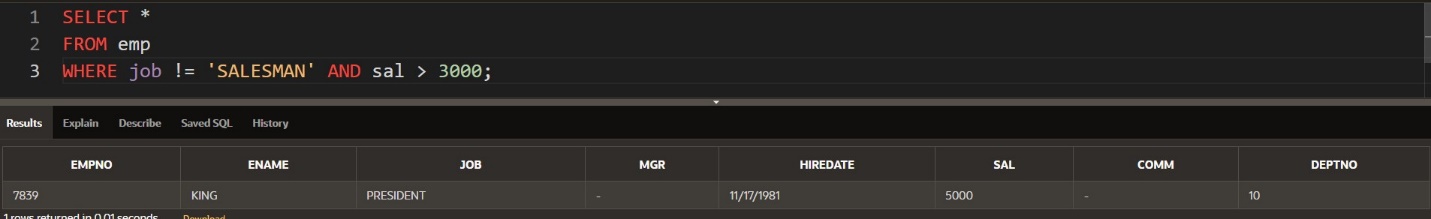
1. Daha sonra fərz eləyin ki, biz istəyirik ki, **SALESMAN** vəzifəsində çalışmayan və maaşı 3000-dən **az** olan işçilərin məlumatlarını əldə etmək istəyirik. Bu zaman biz aşağıdakı şəkildə olan **SQL** query-sini çalışdıraraq istədiyimiz nəticəni əldə edə bilərik.



gördüyünüz kimi burada artıq biz yeni bir operator növündən, yəni, **<(kiçikdir)** operatorundanistifadə etmiş olduq. Burada bu operator mənaca olaraq az olanları mənası kimi başa düşə bilərsiniz.

1. Biz yuxarıdakı şəkildə olan query-də gördük ki, **resultset** də datalar vəzifəcə **SALESMAN** olmayanvə maaşı 3000-dən aşağı olan işçilərin məlumatlarını görmüş olduq. Bəs yaxşı biz istəsək ki, həmçinin 3000 də alan işçi gəlsin **resultset-**mizə bu zaman isə biz aşağıdakı şəkildə olan query-ni yazmış olacayıq.

gördüyünüz kimi yuxarıdakı şəkildə qırmızı ilə qeyd olunan hissə **SALESMAN** olmayan və həmçinin maaş olaraq 3000 də alan işçilərin məlumatlarını əldə etmiş olduq. Bunu biz məhz **<= (less than equals)** operatorunun vasitəsi ilə həyata keçirtmiş olduq.

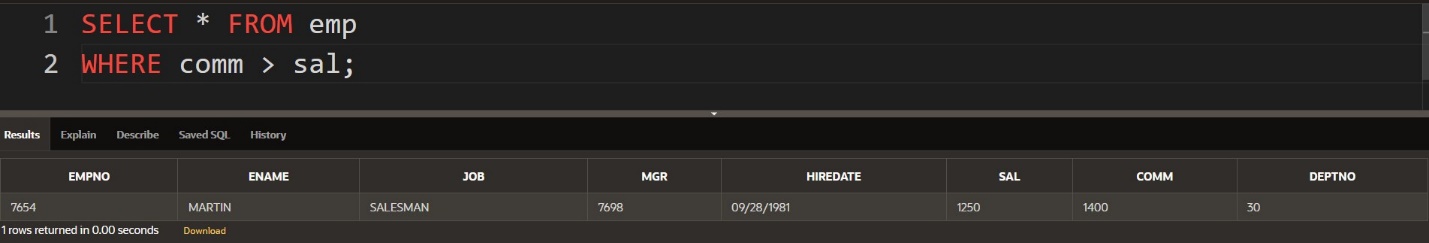
1. Daha sonra məsələn biz istəyirik ki, bu dəfə artıq, **SALESMAN** olmayan və salary-si 3000-dən çox olan işçilərin məlumatlarını əldə edək, bu zaman aşağıdakı şəkildə olan query-ni yaza bilərik.

gördüyünüz kimi, buradada bunu biz **>(böyükdür)** operatorunun vasitəsi ilə həyata keçirtmiş olduq. Yəni çox olan mənasında işlədilməkdədir.

**TASK CHALLANGE**

1. Mənə komisiyası salary-dən çox olan işçilərin məlumatlarını verən query yazmağını istəyirəm.

**TASK SOLVİNG**

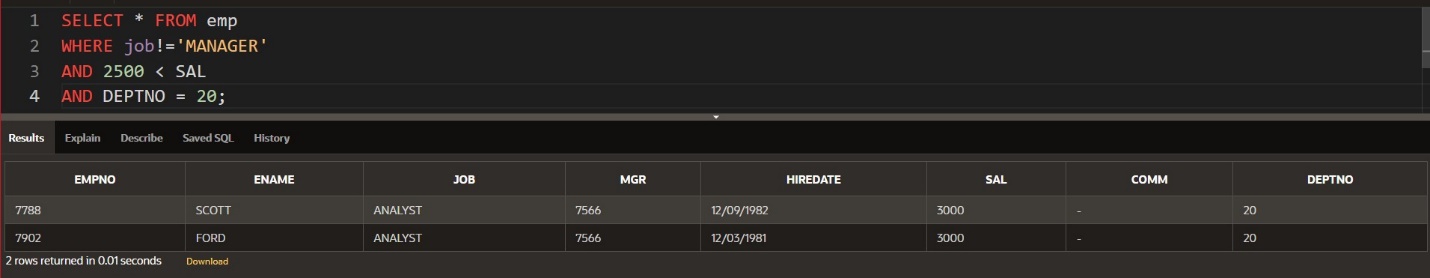
1. Bu taskı həll edə bilmək üçün ilk öncə emp table-ı üzərindən analiz query-sini çalışdırmalısan. Daha sonra aldığın resultset-dən baxmalısanki mən hansı sütunlara görə müqayisə operatorularını istifadə edərək müqayisə əməliyyatını aparacağam. Məhz bu baxımdan taskın solve olunma prosesini həyata keçirtməlisən. Elə isə taskın solve hissəsi aşağıdakı kimidir.

**CHALLENGE TASK**

1. Vəzifəsi MANAGER olmayan, maaşı 2500-dən çox olan və həmçinin 20 nömrəli department də işləyən işçilərin məlumatlarını gətirən query yazın.

**SOLVİNG CHALLENGE**

1. Bu taskın solve hissəsi aşağıdakı şəkildə olan query də verilmişdir.

**Note: WHERE** clause-ında comparsion əməliyyatını həyata keçirdərkən, sol tərəfdə dəyəri sağ tərəfdə isə column adını qeyd etmək worst practice-dir. Bunu aşağıdakı şəkildən görə bilərsiniz.