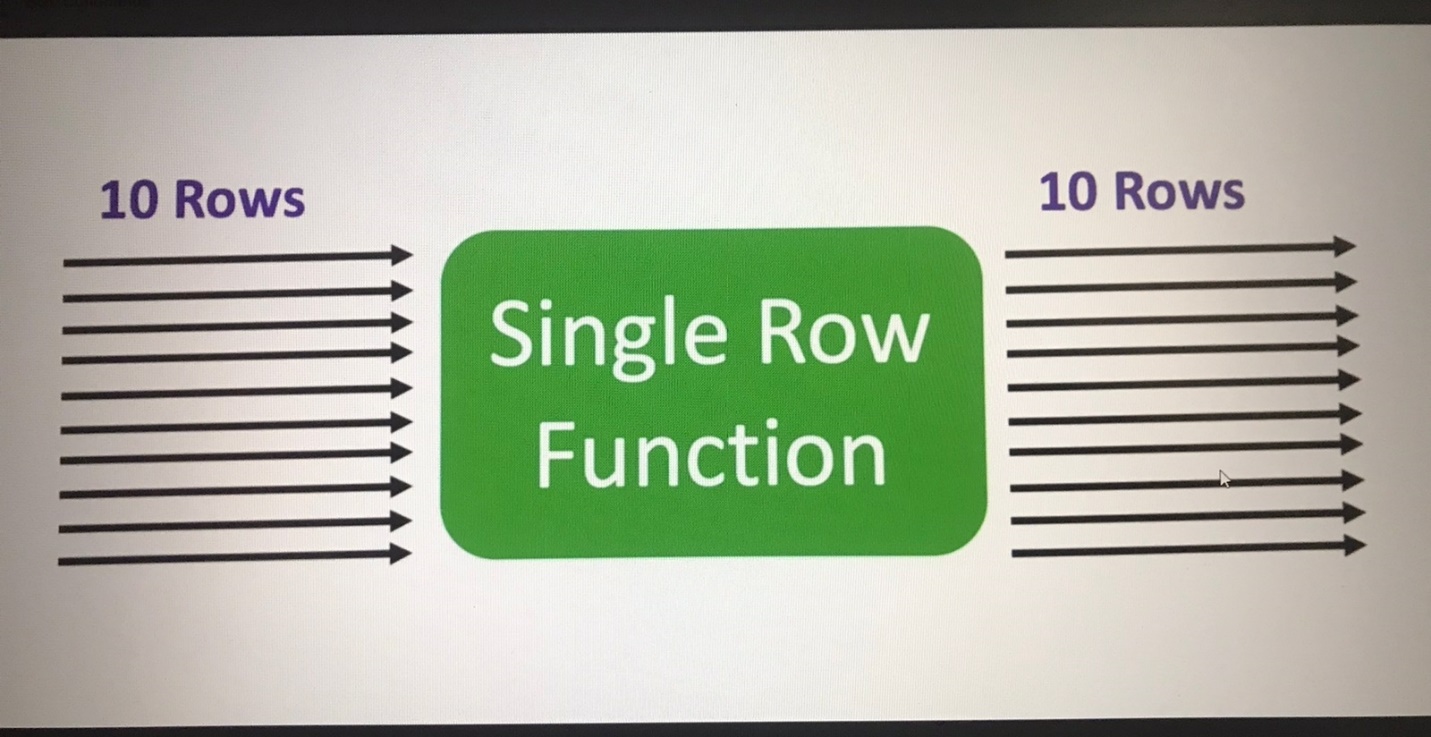
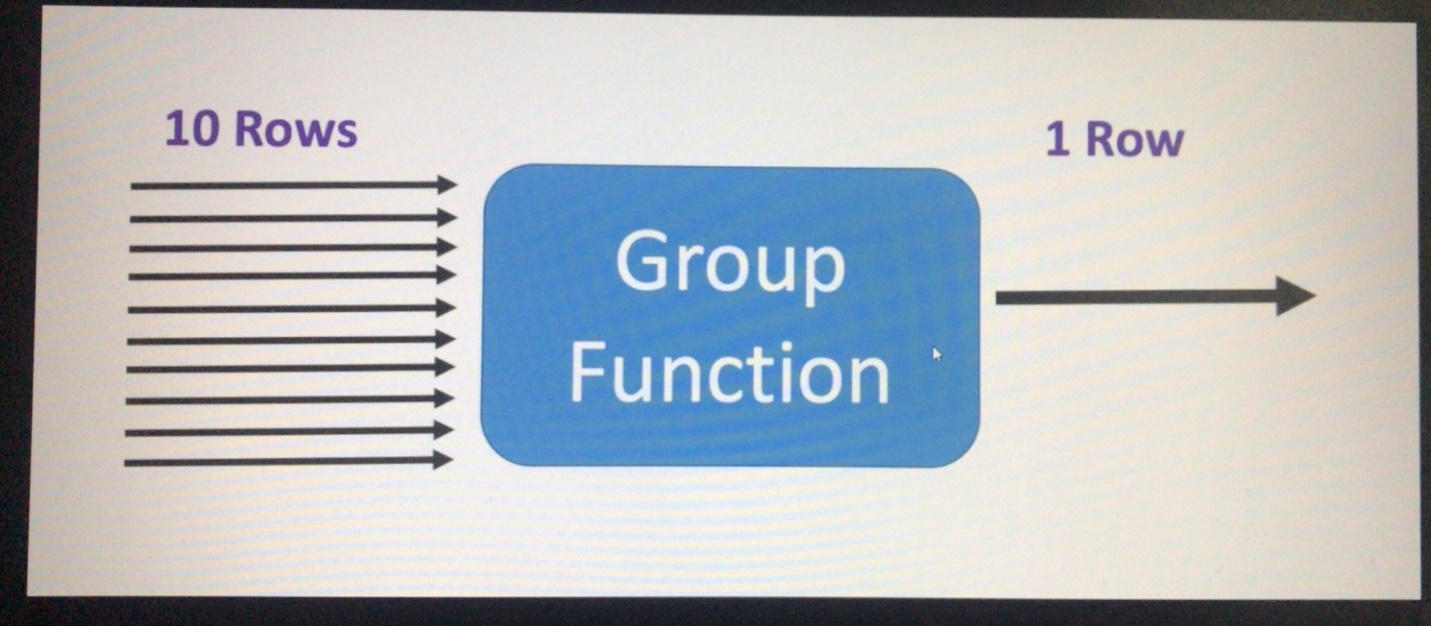
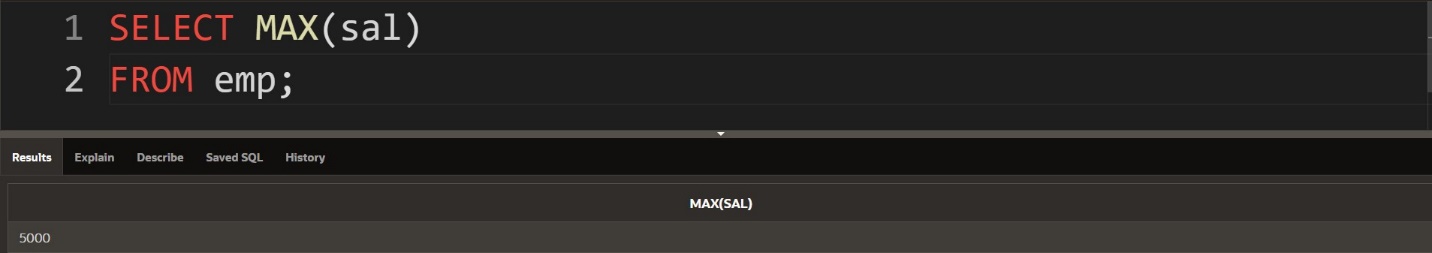
1. Keçən section da biz **SQL** də **SRF-**ların nə olduğunu öyrənmiş olduq və bildik ki, **SRF-**lar bu prinsip ilə işləyir ki, girmiş olduğu **table** üzərində hər bir record-a əsasən özünə aid iş görəcəkdir. Bunu aşağıdakı şəkildən rahat bir şəkildə başa düşə bilərsiniz.

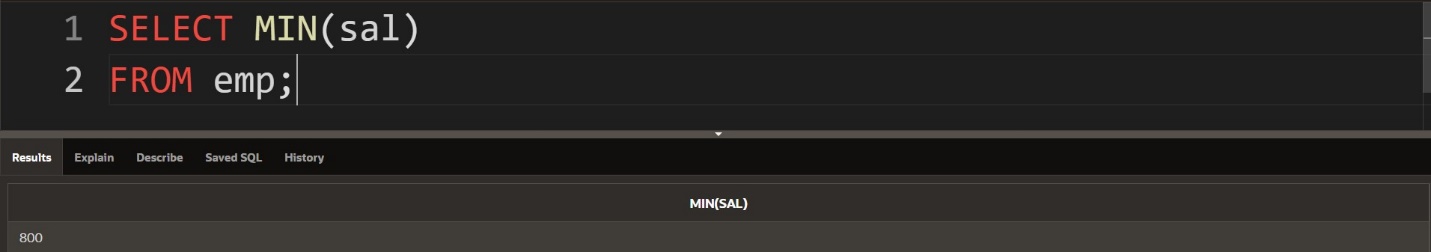
gördüyümüz kimi, yuxarıdakı şəkildə **SRF**-ın işlənmə prinsipi göstərilmişdir, burada belə başa düşmək olar ki, fərz edin bu solda göstərilmiş olan **10 rows** hər hansı bir table da 10 sətirin olduğunu göstərir, **ortada** duran isə bizim hər hansı bir **srf** funskiyasından istifadə etdiyimiz anlamına gəlir və az öncədə dediyimiz kimi **srf** table içinə girərək 10 sətirin hər birində əməliyyat aparıb **sağ** tərəfdə olan **resultset** də bizə o qədər **result** qaytarmış olacaqdır.

1. Bu mövzuda isə biz, **groupping** functions-ların nə olduğunu öyrənəcəyik. Bu funksiyaların **srf** funksiyalardan fərqi ondadır ki, çoxlu input qəbul edib, outputa yalnızca bir nəticə qaytarmaqdadır. Bunu anlamaq üçün aşağıdakı şəkildə göstərilmiş olan **groupping** funksiyaların necə işlədiyini görə bilərsiniz.

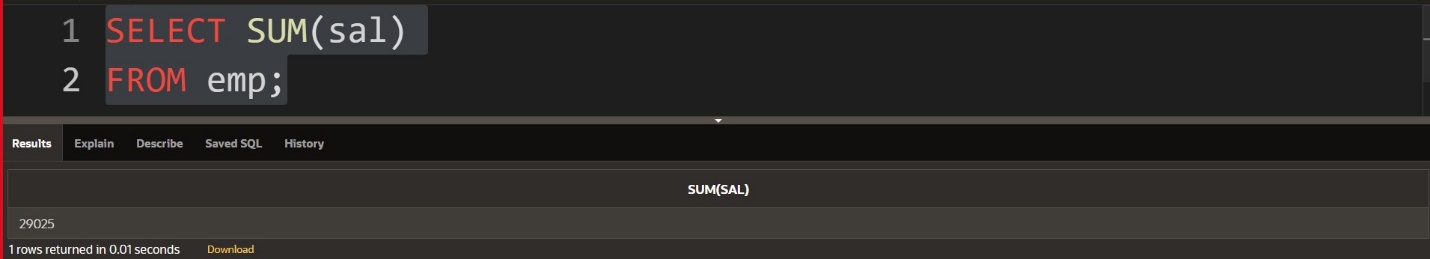
gördüyümüz kimi yuxarıdakı şəkildə **solda** duran 10 rows fərz edin ki, hər hansı bir table da olan 10 sətirə bərabərdir və bu table üzərində biz **groupping** functionlardan istifadə edərək, **ortadakı** şəkil hər hansı bir **groupping** funksiyasının olduğunu göstərməkdədir, beləki istifadə etdiyimiz **groupping** funksiyası özünə aid vəzifəni gördükdən sonra yekun nəticə etibarı ilə **resultset** də tək bir **result** qaytaracaqdır, bax buda **groupping** funksiyalarının **srf** funksiyalarından fərqi.

1. İndi isə gəlin **grouping** funksiyalara aid nümunələrə baxaq. Məsələn gəlin **emp** table-ından ən çox maaş alan işçinin maaş dəyərini ekranda göstərən **query** yazaq. Bunun üçün **SQL** də **MAX()** grouping funksiyası vardır ki, hansıki max dəyəri verməkdədir, eynən aşağıdakı şəkildə olduğu kimi.

gördüyünüz kimi yuxarıdakı query də **MAX** funksiyasının istifadə qaydası göstərilmişdir.

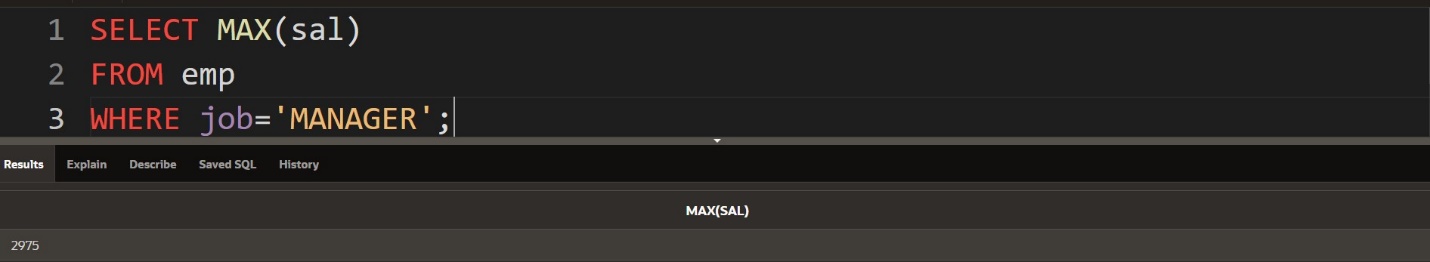
1. İndi isə gəlin **emp** table-ından ən az maaş alan işçinin maaş dəyərini ekranda göstərən **query** yazaq. Bunun üçün **SQL** də **MIN()** grouping funksiyası vardır ki, hansıki max dəyəri verməkdədir, eynən aşağıdakı şəkildə olduğu kimi.

gördüyünüz kimi yuxarıdakı query də **MIN** funksiyasının istifadə qaydası göstərilmişdir.

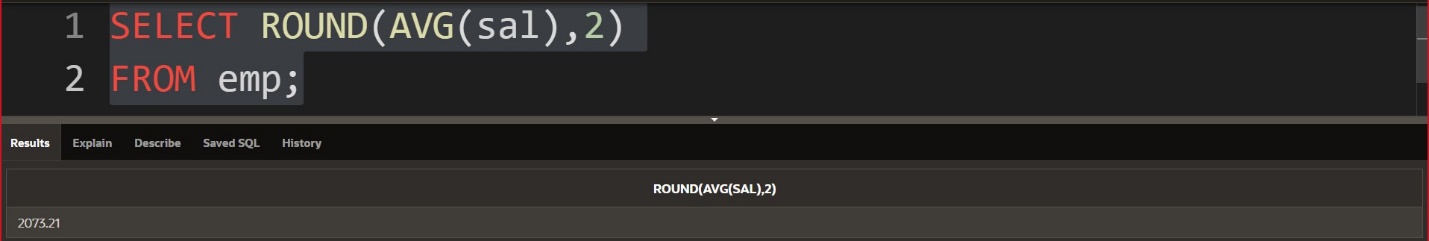
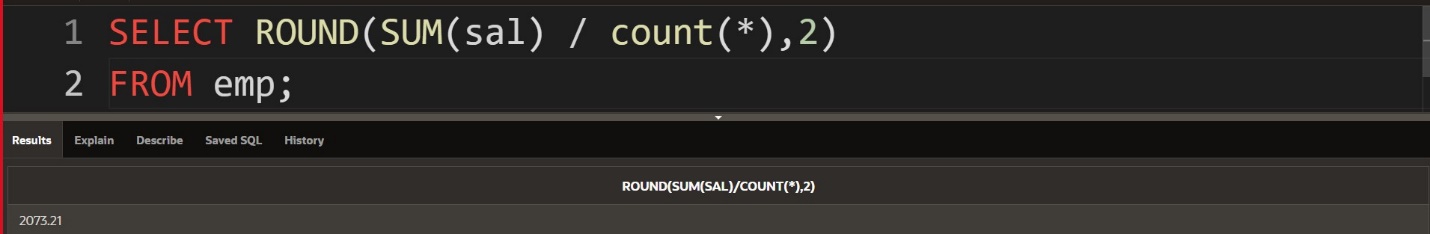
İndi isə gəlin **emp** table-ından bütün işçilərin maaşlarının cəmini hesablıyan query yazaq. Bunun üçün **SQL** də **SUM()** grouping funksiyası vardır ki, hansıki cəmi dəyəri verməkdədir, eynən aşağıdakı şəkildə olduğu kimi.

**TASK CHALLENGE**

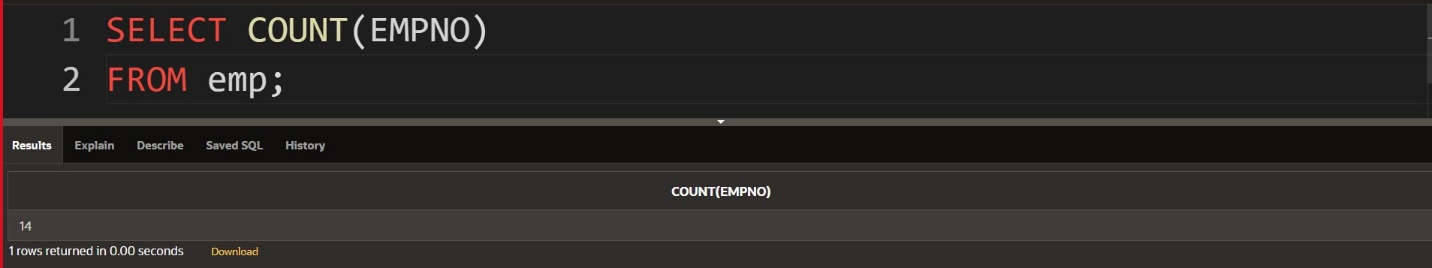
Vəzifəcə **MANAGER** olan və ən yüksək maaş alan **MANAGER-**i tapan query yazın.

**TASK SOLVING**

**AVG() (Ədədi Orta)**

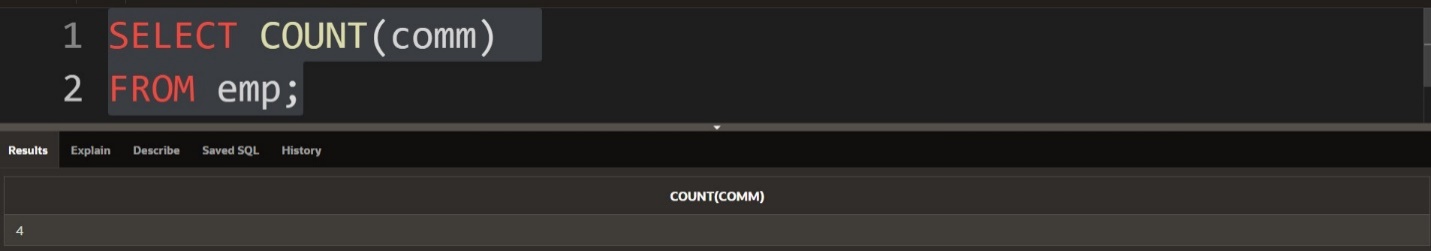
1. Bu funksiya ona verilmiş sütuna əsasən həmən sütundakı dəyərlərin ədədi ortasını tapmaq üçün istifadə olunur. Məsələn aşağıdakı şəkildə olduğu kimi.
2. Həmçinin biz **AVG()** funksiyasının dolayı yol ilədə yazılışını aşağıdakı şəkildən görə bilərik.

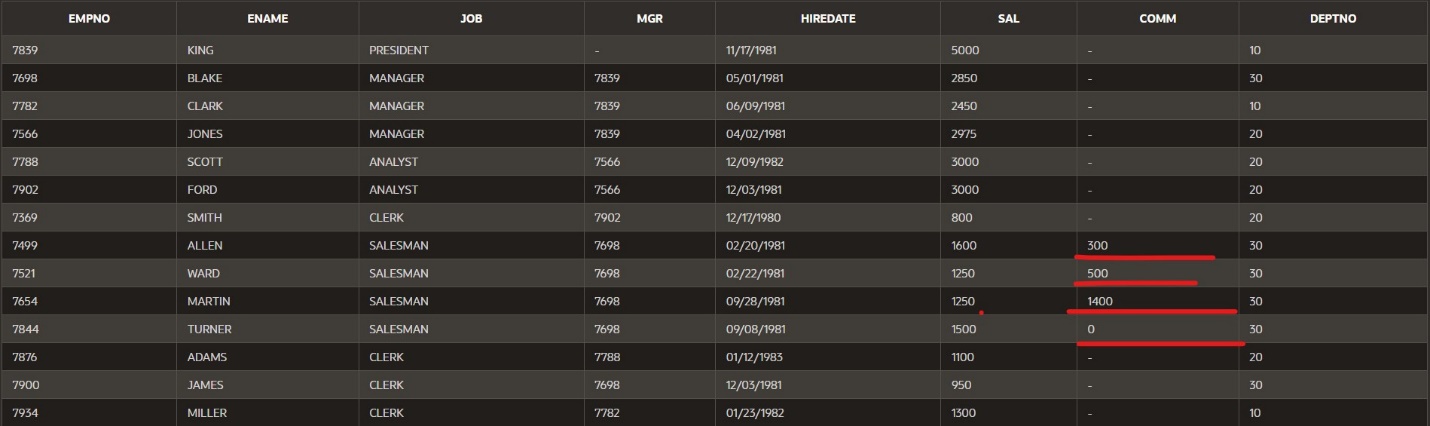
**COUNT();**

1. Bu funksiya ona verilmiş sütuna əsasən həmən sütunda olan dəyər neçə sətirdə vardırsa onların sayını qaytaracaqdır, eynən aşağıdakı şəkildə olduğu kimi.

gördüyünüz kimi yuxarıdakı şəkildə biz **emp** table-ından neçə ədəd sətirdə **empno** sütununda olan dəyər mövcuddur.

* **NOTE:** Əgər sütunda **NULL** dəyəri vardırsa və onu siz **COUNT()** funksiyasına vermiş olsanız, bu zaman həmən **NULL** dəyərlər hesaba qatılmıyacaqdır, eynən aşağıdakı şəkildə olduğu kimi.

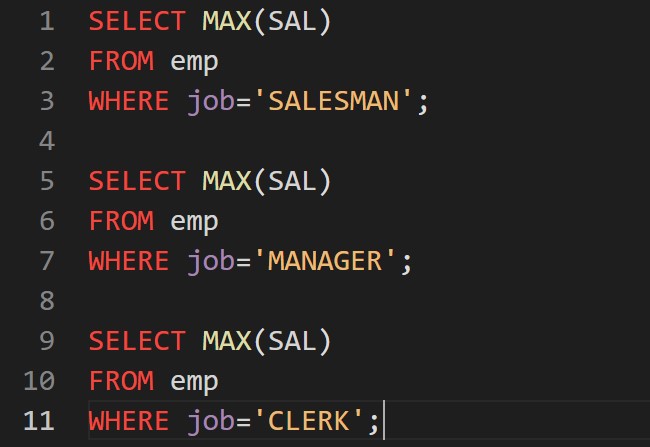
****

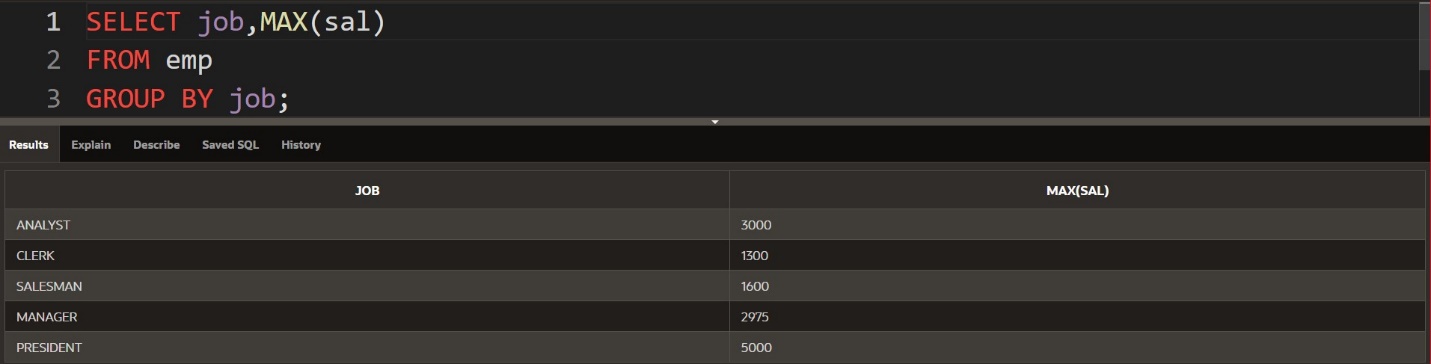
gördüyünüz kimi, **comm** sütununa görə **COUNT()** funskiyasın çağırıb işə saldıqda bizə **resultset** də 4 dəyərin qaytaracaqdır, bu da o deməkdir ki, **comm** sütununda olan dəyərlər cəmi-cümlətan 4 sətirdə mövcuddur, aşağıdakı şəkildə olduğu kimi.

* **NOTE: COUNT()** funksiyasın məsləhət görülür ki, hər hansı bir sütun adın parameter olaraq ötürmək əvəzinə oraya **\*(asterisk)** operatorunu ötürmək lazımdır. Buna səbəb odurki, **NULL** dəyərlər **COUNT()** funksiyasında hesaba qatılmadığından **best practise** olaraq **\*(asterisk)** operatorunu ötürmək lazımdır.

**GROUP BY**

1. Bu keyword bizə **SQL** də o şəraiti yaradır ki, **SRF**-da olduğu kimi bir çox sətirdə olan **result** çıxarda bilək, ancaq gördük ki, **groupping** functionlar sadəcə geriyə **resultset** də bir **result** qaytarır.
2. Bizə o şəraiti yaradır ki, ayrı-ayrı çox saylı query-lər yazıb aggregate function-ları sütundakı fərqli dəyərlərə görə test etmək əvəzinə elə bir query yazıb sütundakı fərqli-fərqli dəyərlərə görə test edə bilək. Məsələn bu son yazdığımı daha yaxşı anlaya bilmək üçün aşağıdakı şəkildə olan query-ə baxaq görək nə demək istəyirik.



gördüyünüz kimi, biz **job** sütununda olan fərqli-fərqli vəzifədə işləyən və ən çox maaş alan işçinin məlumatlarını öyrənmək üçün belə ayrı-ayrı query yazmış olduq. Tamam deyə bilərsizki belədə işliyəcək, bəs yaxşı fərz edin ki, **job** sütununda 100-lərcə hətta 1000 cürnə vəzifədə çalışan işçi vardır, o zaman gedib nə edəcəksiniz, 1000-dənəsi üçün də ayrı-ayrılıqda yuxarıdakı kimi query yazacaqsınız? Bu problemi həll edə bilmək üçün biz bunu **SQL** də **GROUP BY** clause-ından istifadə edərək, onları qruplaşdıraraq onlar haqqında istifadə etdiyimiz hər bir **aggregate** funksiyasına görə dəyərləri ala bilirik, eynən aşağıdakı şəkildə olduğu kimi.