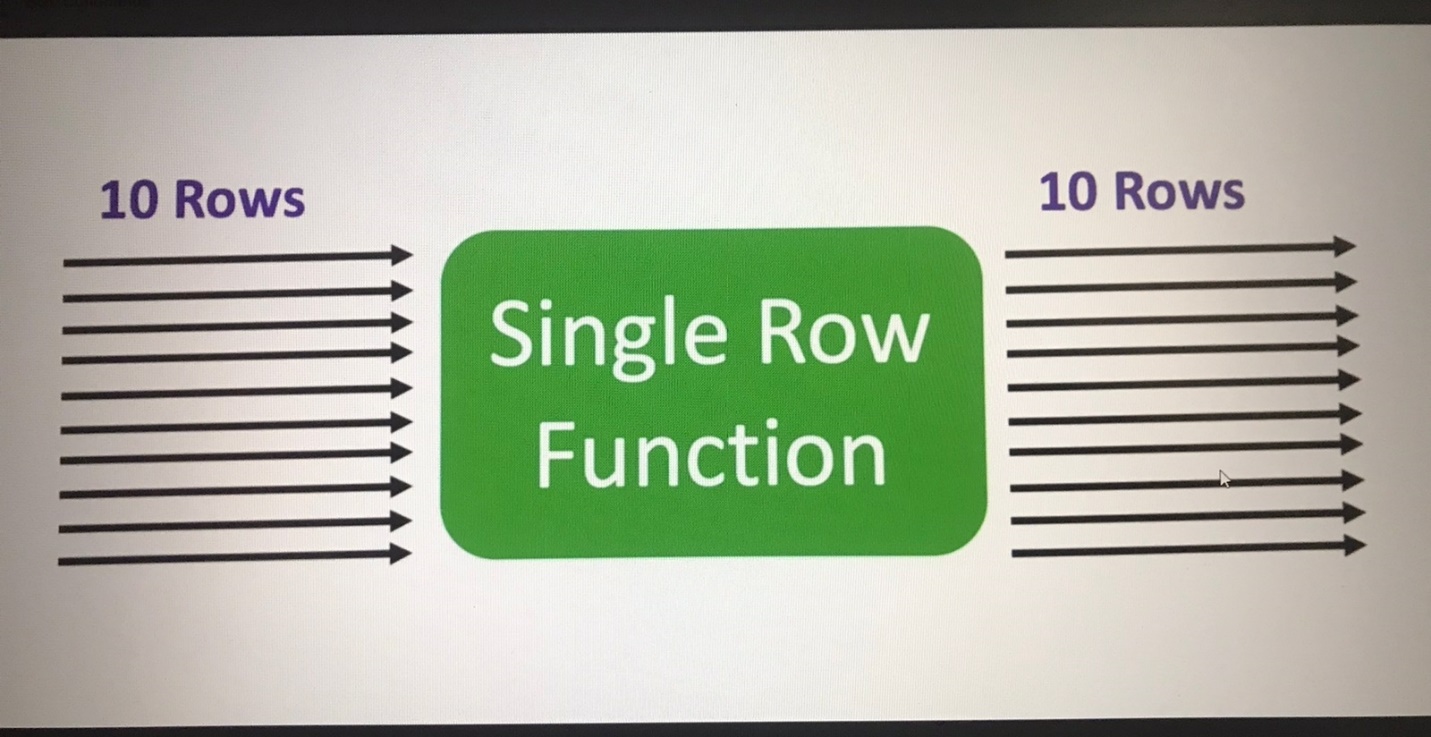
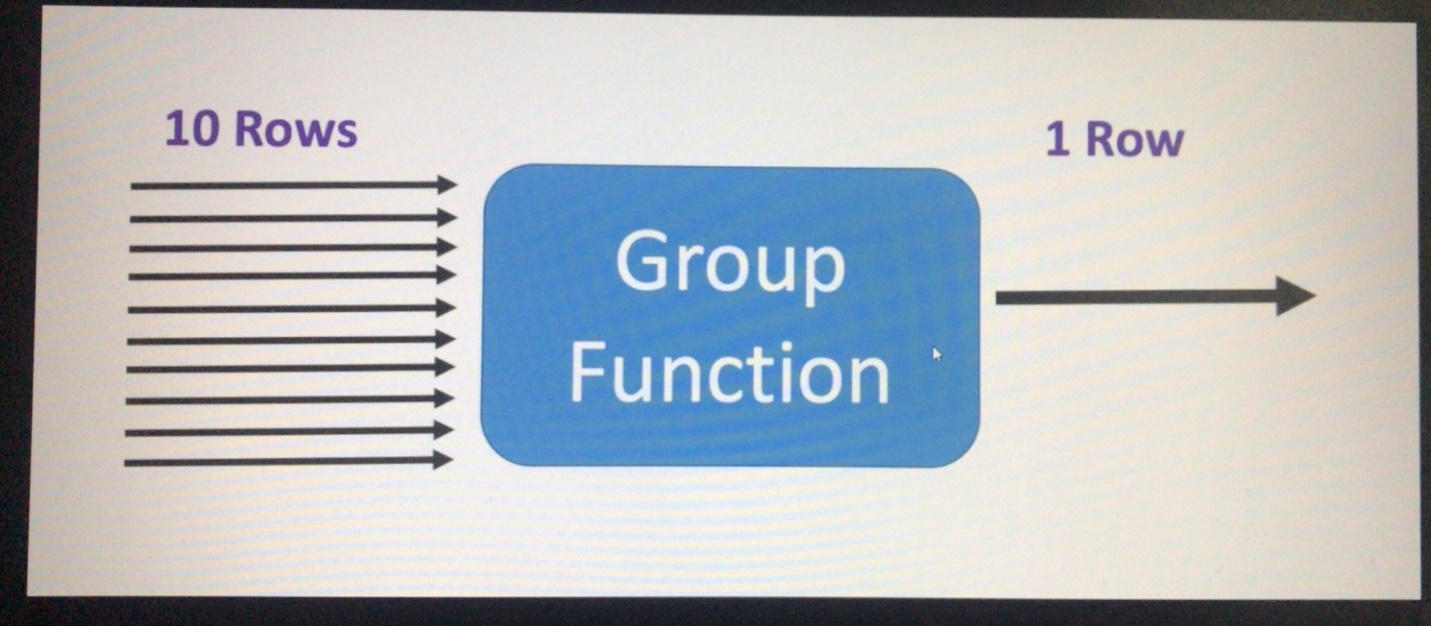
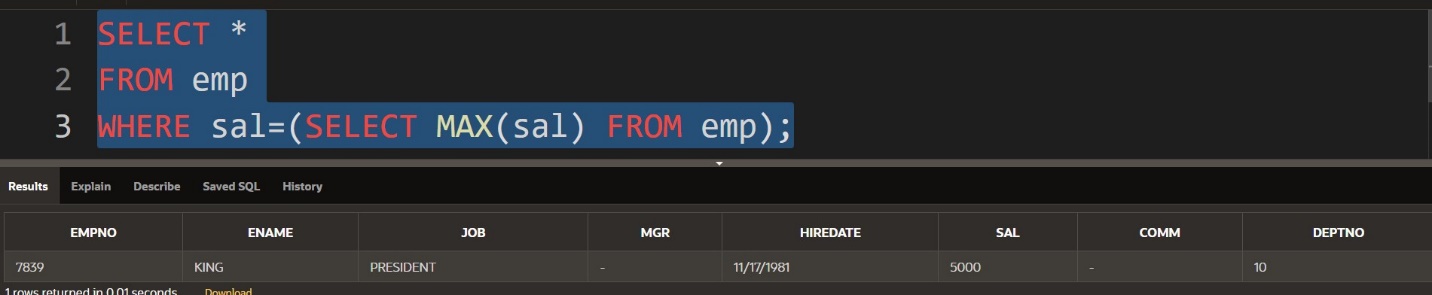
1. Keçən section da biz **SQL** də **SRF-**ların nə olduğunu öyrənmiş olduq və bildik ki, **SRF-**lar bu prinsip ilə işləyir ki, girmiş olduğu **table** üzərində hər bir record-a əsasən özünə aid iş görəcəkdir. Bunu aşağıdakı şəkildən rahat bir şəkildə başa düşə bilərsiniz.

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 göstərmək deməkdir, **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. İndi isə öyrənəcəyimiz olan bu dərslikdə yəni, bizdə adlanacaqdır, **Groupping** functions-lar. Bu tip funksiyaların **srf**-lardan fərqi ondadır ki, onlar girmiş olduğu table daxilində verilmiş şərtə əsasən ən üst səviyyə hansıdır onu seçib **resultset** də qaytarır, daha **srf**-larda olduğu kimi bütün dəyərləri vermir, yalnızca bir dəyər verir, eynən aşağıdakı şəkildə olduğu kimi.

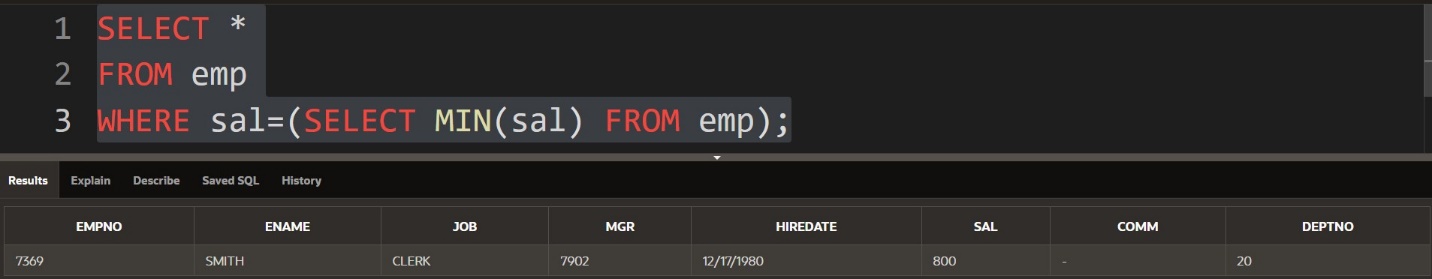
gördüyümüz kimi yuxarıdakı şəkildə **solda** duran 10 rows fərz edin ki, hər hansı bir table da olan 10 sətirə bərabərdir və bu table üzərində biz groupping functionlardan istifadə edərək, **ortadakı** şəkildəki kimi, istifadə etdiyimiz groupping funksiyası özünə aid vəzifəni gördükdən sonra yekun nəticə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

**MAX();**

1. **MAX()** funksiyası ona verilmiş sütun-a görə ən böyük dəyərə malik olan dəyəri tapıb çıxardır **resultset-**ə hər hansı bir query apardığınız table-dan, eynən aşağıdakı şəkildə olduğu kimi.

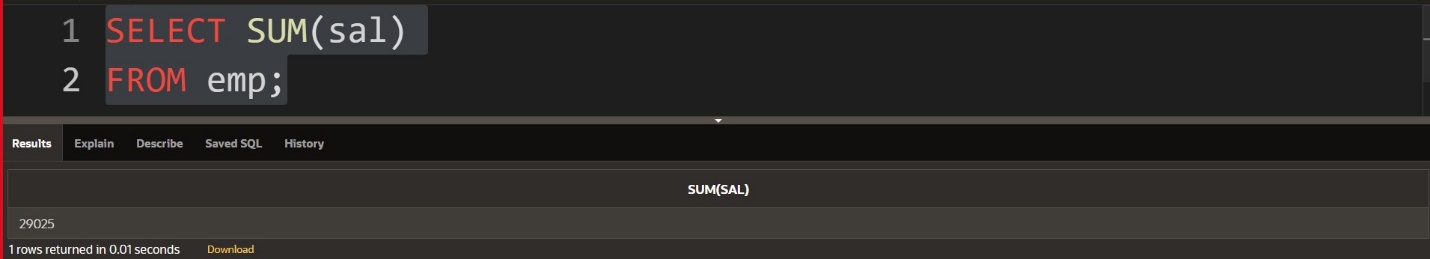
gördüyünüz kimi yuxarıda olan şəkildə biz **emp** table-ından ən çox maaş nə qədərdir onu öyrənmiş olduq.

**MIN();**

1. **MAX();** əksi olaraq **MIN();** isə ən az dəyərə sahib olan **result** çıxardır, eynən aşağıdakı şəkildə olduğu kimi.

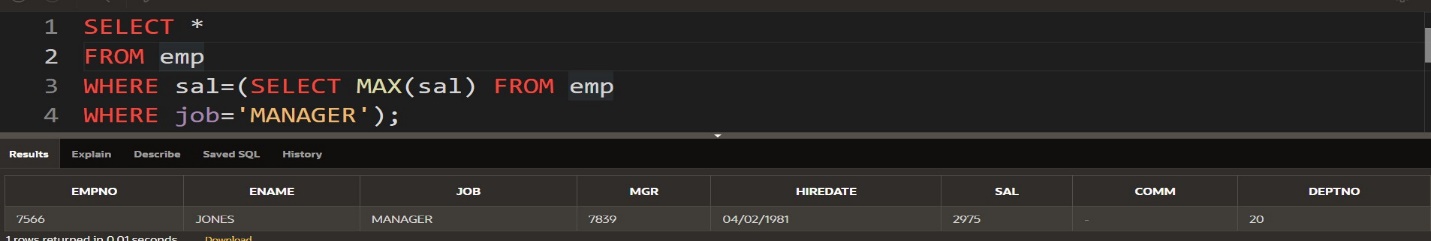
gördüyünüz kimi **emp** table-ından ən az maaşa malik olan dəyəri **MIN();** funksiyasının köməyi ilə əldə etmiş olduq.

**SUM();**

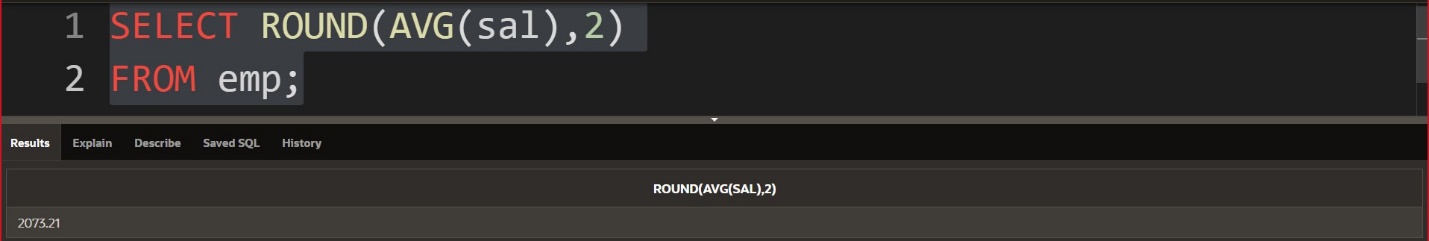
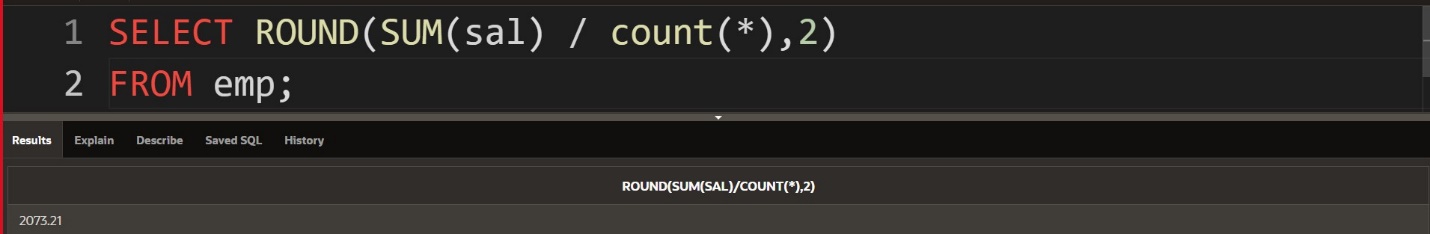
1. ****Bu funksiya isə ona verilmiş olan sütuna əsasən sütundakı bütün dəyərlərin **cəmini** qaytarmaqdadır, eynən aşağıdakı şəkildə olduğu kimi.

**TASK CHALLENGE**

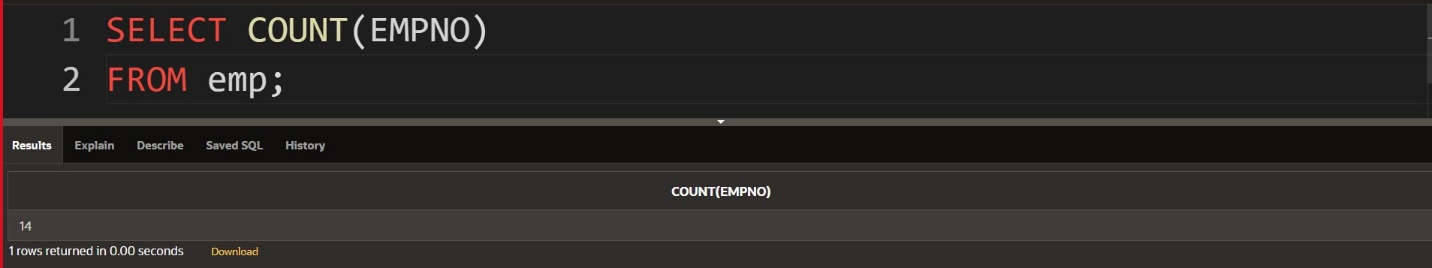
Vəzifəcə **MANAGER** olan və ən yüksək maaş alan **MANAGER-**i tapan query yazın.

**** **TASK SOLVING**

**AVG() (Ədədi Orta)**

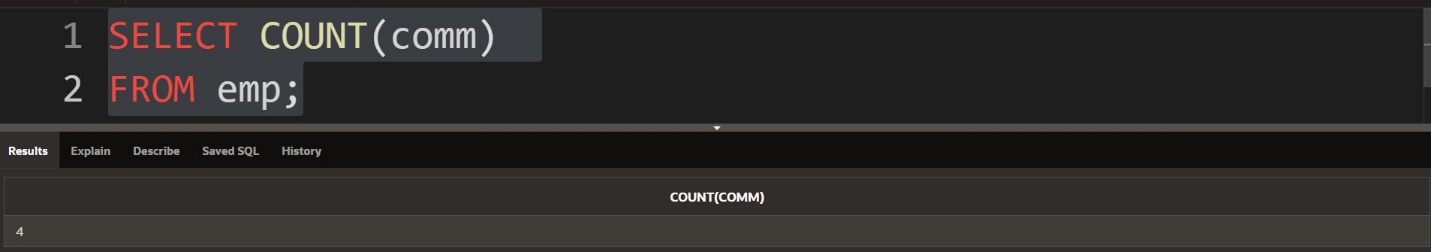
1. Bu funksiya ona verilmiş sütuna əsasən həmən sütundakı dəyərlərin ədədi ortasını tapmaq üçün istifadə olunur. Məsələn aşağıdakı şəkildə olduğu kimi.
2. Həmçinin biz **AVG()** funksiyasının dolayı yol ilədə yazılışını aşağıdakı şəkildən görə bilərik.

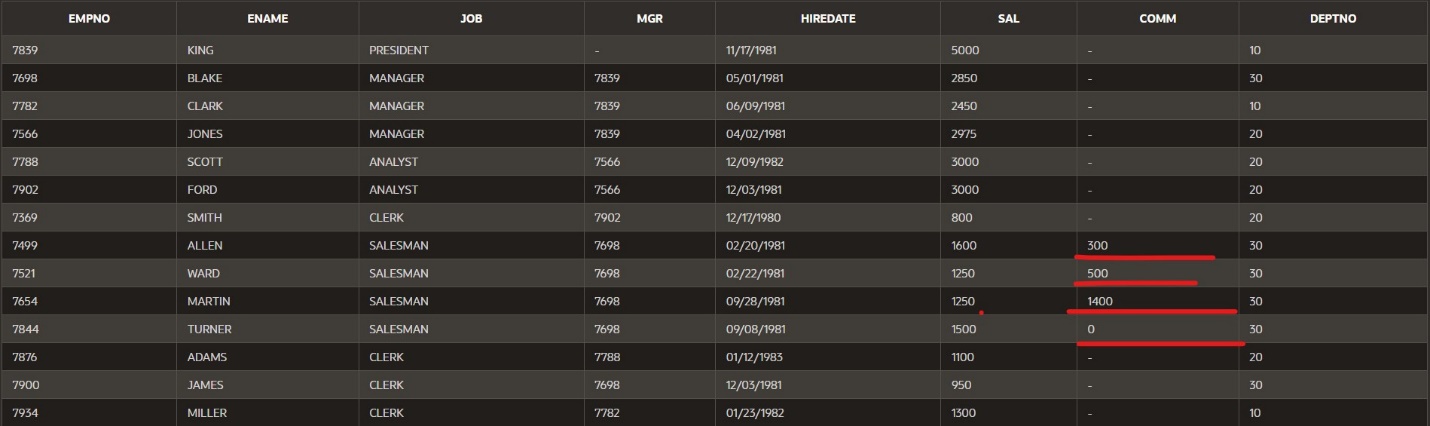
**COUNT();**

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.

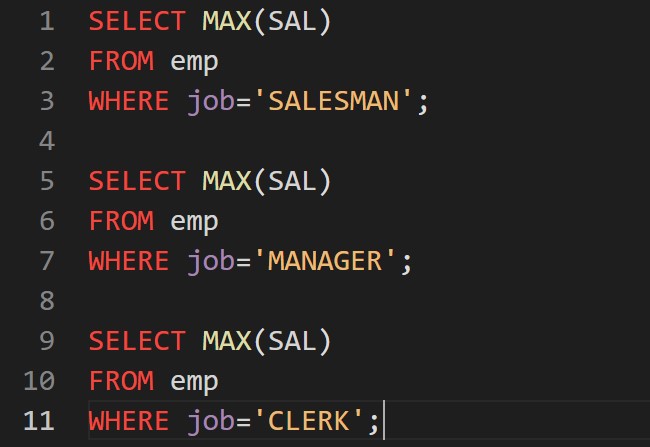
****

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, ola bilər ki, siz hər hansı bir table-da neçə record(row,sətir) olduğunu bilmək istəyərsiniz.

**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-mi 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.

