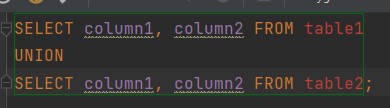
Bu mövzudan etibarən **SQL** də **set** operatorlarını öyrənmiş olacayıq. Əksər databazalarda məlumatlar fərqli-fərqli table-larda saxlanılmaqdadır və çox zaman biz həmən bu fərqli-fərqli table-lardan məlumatları birləşdirib eyni anda çəkmək istəyə bilərik. **SQL,** bizə fərqli-fərqli table-lardan məlumatları eyni anda çəkib birlikdə çıxartmağımıza şərait yaradan **set** operatorlarını təqdim edir . **Set** operatorları aşağıdakılardır:

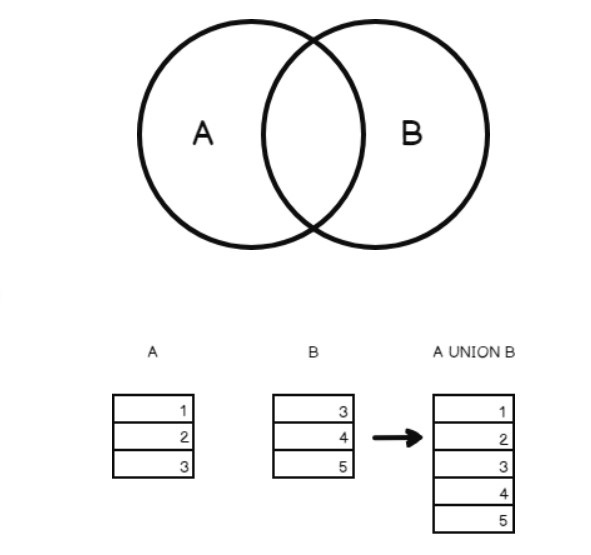
1. **UNION/ALL**
2. **INTERSECT/ALL**
3. **EXCEPT/ALL**

**UNION**

1. İki və ya daha çox fərqli table-lardan qayıdan sətirləri bir **resultset** də birləşdirmək üçün istifadə olunan opeartor **UNION/ALL** adlanır. Bu operator iki və ya daha çox table-dan qayıdan uyğun gələn sütunlar üzərə sətirləri biryerdə birləşdirib qaytarmaqdadır. Gəlin elə isə aşağıdakı sintaksisə baxaraq nə olduğunu anlamağa çalışaq.

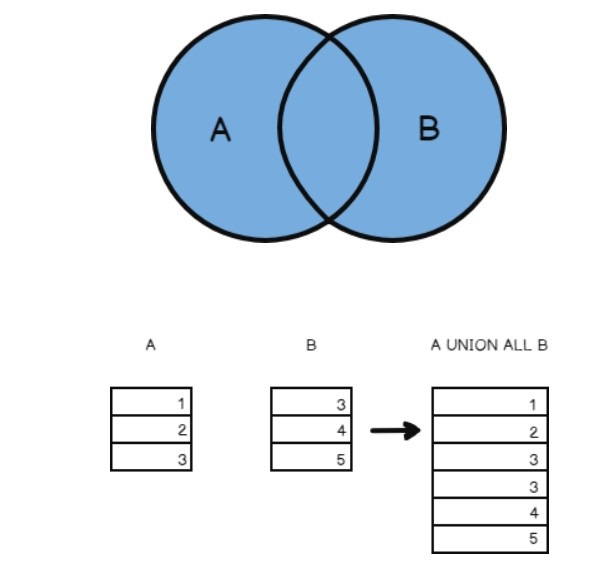


yuxarıdakı şəkildə olan query də **UNION** operatoru databazaya hər bir **SELECT** statement-in ayrı-ayrılıqda çalışmasını və onlardan qayıdan nəticəni bir **resultset** table-ında birləşdirməsini deməkdədir. Həmçinin yuxarıdakı şəkildə olan query tək bir query sayılmaqdadır, çünki ilk **SELECT** statement-ı “**;”** bitməmişdir.

1. Aşağıdakı şəkildə **UNION** nədir onun flowchartı göstərilmişdir.

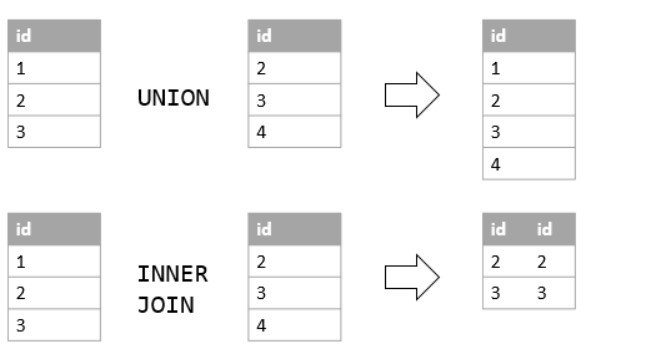
yuxarıdakı şəkildə gördüyümüz kimi **A** table-ı 1,2,3 və **B** table-ı isə 3,4,5 dəyərlərini saxlamaqdadır. Nəticə olaraq **A** və **B** table-ları **UNION** olunaraq bir **resultset** table-ında saxlanılmaqdadır, lakin fikir versəniz hər iki table-da da təkrar dəyərlər var, lakin onlar **resultset** table-ında unikal şəkildə saxlanılmaqdadır, buna səbəb **UNION**-ınyalnızca hər iki table-danda unikal sətirləri birləşdirməsidir.

**UNION ALL**

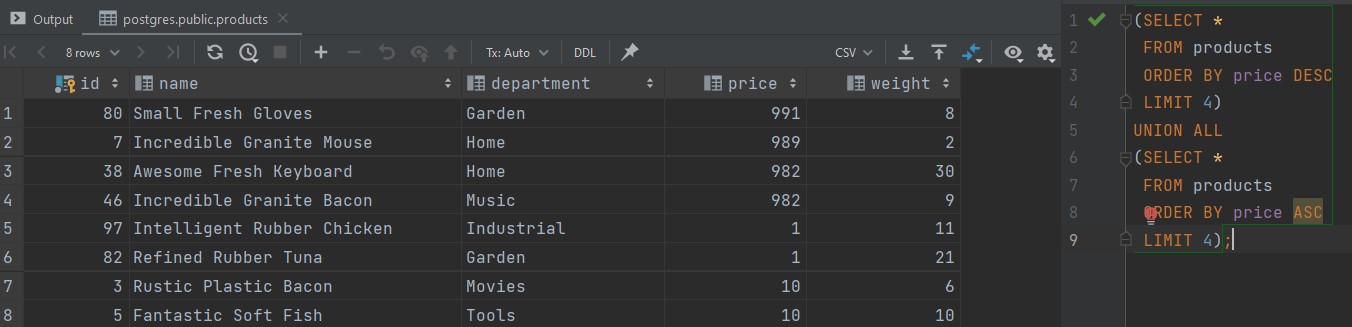
**UNION** və **UNION ALL** demək olarki eynişeydir, lakin onlar arasında performans cəhətindən önəmli fərqliliklər vardır. **UNION ALL** da **UNION** kimi iki və ya daha çox fərqli table-lardan sətirdəki məlumatları birləşdirərək onları bir **resultset** də çıxartmağımıza şərait yaradır, lakin **UNION**-dan fərqli olaraq o **duplicate** sətirdəki dataları saxlamaqdadır.

yuxarıdakı şəkildə gördüyümüz kimi **A** table-ı 1,2,3 və **B** table-ı isə 3,4,5 dəyərlərini saxlamaqdadır. Nəticə olaraq **A** və **B** table-ları **UNION** olunaraq bir **resultset** table-ında saxlanılmaqdadır, lakin fikir versəniz hər iki table-da da təkrar dəyərlər var və onlar **resultset** table-ında da saxlanılmaqdadır, buna səbəb **UNION ALL**-ın duplicate sətirdəkidataları da birləşdirə bilməsidir.

**UNION vs JOIN**

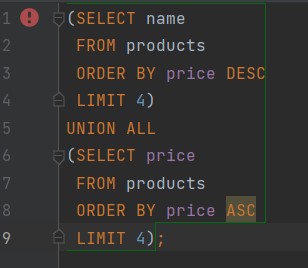
**JOIN**-lərdən məsələn **INNER JOIN** və yaxud **LEFT JOIN** iki table-dan column-ları birləşdirərkən, **UNION/ALL** isəiki querydən qayıdan sətirləri birləşdirməkdədir. Başqa deyimlə desək, **JOIN** nəticələri horizontal yəni yanaşı bitişdirərkən, **UNION/ALL** isə vertical yəni yuxarıdan aşağı şəklində bitişdirir nəticələri,eynən aşağıdakı şəkildə olduğu kimi.

1. İndi isə gəlin bir query yazmağa çalışaq görək nə olan şeydir **UNION**. Gəlin 4 ən bahalı və ən ucuz **product**-ları göstərən query yazmağa çalışaq.

yuxarıdakı şəkildə olan query də gördüyünüz kimi biz 4 ən bahalı və 4 ən ucuz **product**-ları gətirmiş olduq və bunu **UNION ALL** set operatoru vasitəsi ilə həyata keçirtdik. Gördüyümüz kimi, biz 2 fərqli **SELECT** statement yazaraq eyni anda qayıdan nəticələri bir **resultset** də birləşdirmiş olduq.

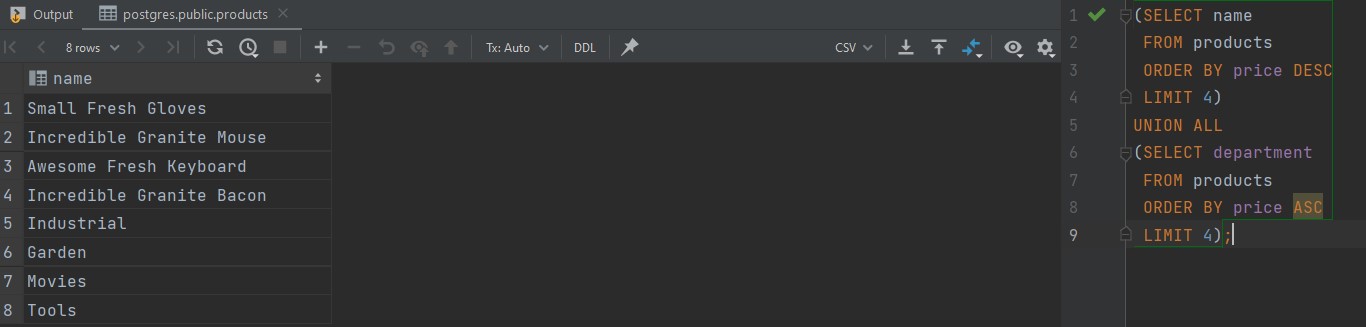
* **NOTE:** Set operatorlarından istifadə edərkən bilinməsi lazım olan bəzi məqamlar vardır.

1. **UNION** edəcəyiniz **SELECT** statementin da **column type**-lar eyni olmalıdır,əks halda error ilə qarşılaşacaqsınız, eynən aşağıdakı şəkildə olduğu kimi.

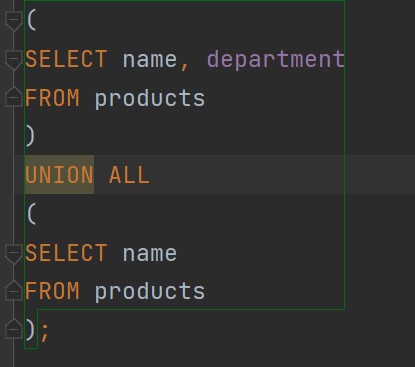




gördüyünüz kimi, yuxarıdakı query də birinci **SELECT** də **name**, ikinci **SELECT** də isə **price** sütunları kimi fərqli tipdə sütunları çəkmək istədiyimizdən xəta ilə qarşılaşdıq.

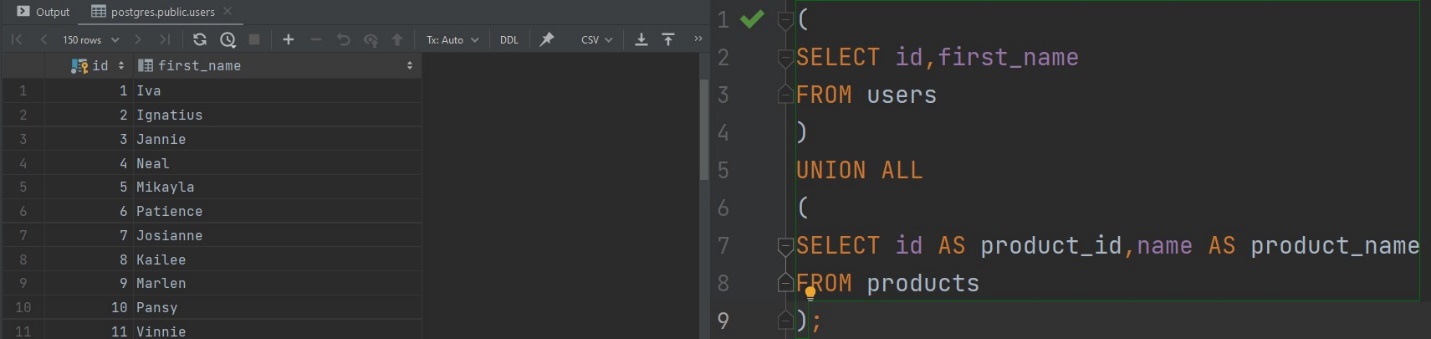
1. column type-lar eyni lakin fərqli sütunlara görə də set operatorlarından istifadə edə bilərsiniz, eynən aşağıdakı şəkildə olduğu kimi.

gördüyünüz kimi, yuxarıdakı query də biz birinci **SELECT** də **name**, ikinci **SELECT** də isə **department** sütunlarını **UNION** edərək heç bir xəta almadan nəticəni almış olduq.

1. **UNION** uniqueness dataları gətirərkən, **UNION ALL** duplicate-ləridə gətirəcəkdir.
2. **column**-ların sayı eyni olmalıdır hər iki querydə də əks halda xəta ilə qarşılaşacayıq, eynən aşağıdakı şəkildə olduğu kimi.

gördüyümüz kimi yuxarıdakı query də ilk **SELECT** statement də 2 ədəd, digər **SELECT** statement də 1 ədəd **column** əlavə etdiyimizdən xəta ilə qarşılaşdıq.

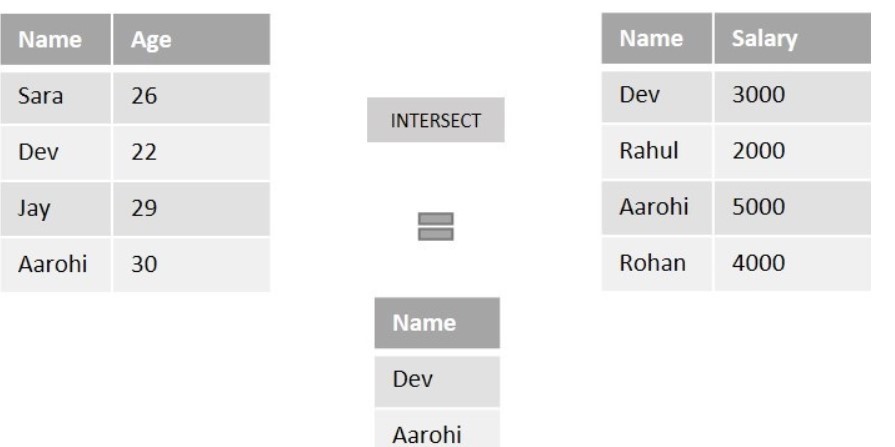
* Qısası, tipləri və sayları eyni olmalıdır.

1. Əgər siz **set** operatorlarından istifadə edərkən, columnlardan hər hansı birinə **alias** versəniz belə birinci **SELECT** hissəsində column adları nədirsə həmən adlara görə **resultset** table-ında sütunları görmüş olacaqsız, eynən aşağıdakı şəkildə olduğu kimi.

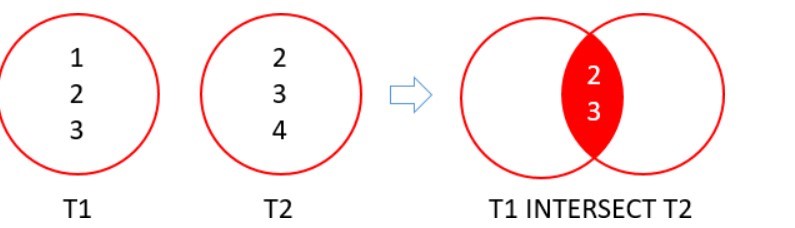
gördüyünüz kimi ikinci **SELECT** hissəsində baxmıyaraq ki, **alias**-lardan istifadə etmişik, lakin yenədə ilk **SELECT** hissəsində sütun adları nədirsə həmən sütun adları çıxmaqdadır.

* **UNION ALL**, **UNION**-dan performans baxımından daha sürətlidir, buna səbəb **UNION ALL**-ın duplicate dataları **UNION** kimi filterləməməsidir.
* **Set** operatorlarından yalnızca **UNION ALL** duplicate-ləri götürə bilməkdədir, yerdə qalan digərləri duplicate-ləri ignore eləyir.
* Əgər əminsinizsəki table-larınızda duplicate sətirlər olmuyacaqdır o zaman **UNION ALL** istifadə edirik ki, performans baxmından sürətli olsun, yox əgər əmin deyilsinizsə **UNION** istifadə etməyiniz tövsiyə olunur.

**INTERSECT/ALL**

Bu **set** operatoru isə iki və ya daha çox table-ın hər ikisindədə ortax məxrəc olan sətirdəki məlumatları gətirmə işinə baxacaqdır. Əgər bir query-də olan **result**,digər query-də yoxdursa, həmən olmuyan-lar ortadan götüzdürləcəkdir, yalnızca biri-birinin qarşılığı olan sətirdəki məlumatları **resultset**-ə salınacaqdır. Bunu daha yaxşı anlamaq üçün gəlin aşağıdakı şəkilə baxaq.

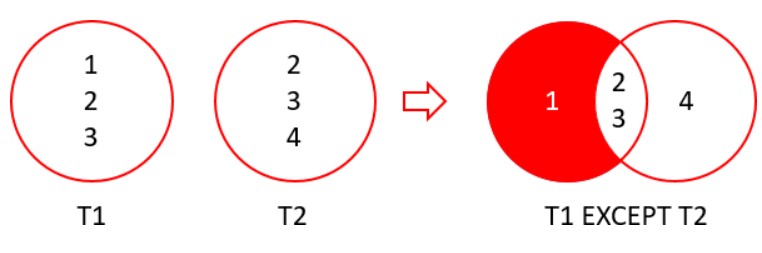
yuxarıdakı şəkildəndən də gördüyünüz kimi hər ikitable-ı biri-birinə **INTERSECT** edərək yalnızca hər iki table-da olan ortax məxrəcə uyğun gələn sətirdəki məlumatları **resultset** də bitişdirərək saxlamış olduq.



yuxarıdakı şəkildəndə gördüyünüz kimi, **T1** və **T2** table-larında yalnızca hər iki tableda biri-birinin qarşılığı olan sətirlərdəki məlumatlar **resultset**-ə daxil olundu, digərləri isə aradan götürüldü.

1. **UNION/ALL** da olduğu kimi, buradada **column** tipləri və ardıcıllığı hər iki **INTERSECT** olunacaq query də eyni olmalıdır.

**EXCEPT/ALL**

**EXCEPT** set operatoru ondan əvvəl yazılan **SELECT** statement-indən qayıdan nəticə, ondan sonra yazılan **SELECT** statement-indən qayıdan nəticədə olmazsa, həmən əvvəl yazılan **SELECT** statement-indən qayıdan nəticə **resultset**-ə daxil olunur. Bunu daha yaxşı anlamaq üçün gəlin aşağıdakı nümunələrə baxaq.

yuxarıkdakı şəkildə gördüyünüz kimi **T1** və **T2** çoxluqları göstərilmişdir və bu **T1** və **T2** çoxluqları **EXCEPT** olunaraq **T1** də olanlar **T2** də yoxdursa, məhz həmən **T1** də olanlar **resultset**-ə yığılsın izahı göstərilmişdir. **T2** də olanlar və olmuyanlar **T1** də varsa və yaxud yoxdursa onlar ignore olunsun izahı göstərilmişdir.

* **NOTE: SET** operatorlarında **ORDER BY** operatorunu yalnızca ən sonuncu **SELECT** hissəsində yaza bilərsiniz və **ORDER BY** edəcəyiniz sütun isə ilk **SELECT** hissəsində olan **sütun** adına görə olmalıdır.