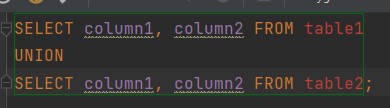
Bu mövzudan etibarən **SQL** də **set** operatorlarını öyrənmiş olacayıq. Əksər databazalarda məlumatlar fərqli-fərqli table-larda saxlanılmaqdadır və çox zaman biz həmən bu fərqli-fərqli table-lardan məlumatları birləşdirib eyni anda çəkmək istəyə bilərik. **SQL** bizə fərqli-fərqli table-lardan məlumatları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

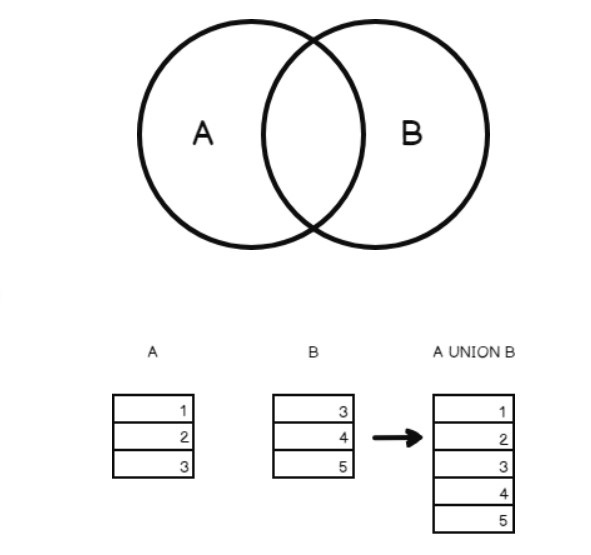
1. **UNION/ALL**
2. **INTERSECT/ALL**
3. **EXCEPT/ALL**

**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.

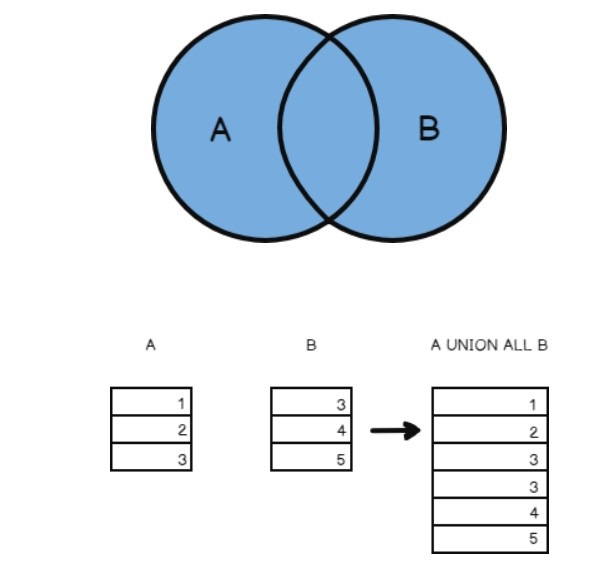


yuxarıdakı şəkildə olan query də **UNION** operatoru databazaya hər bir **SELECT** statement-in ayrı-ayrılıqda çalışmasını və onlardan qayıdan nəticəni bir **resultset** table-ında birləşdirməsini deməkdədir. Həmçinin yuxarıdakı şəkildə olan query tək bir query sayılmaqdadır, çünki ilk **SELECT** statement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

1. Aşağıdakı şəkildə **UNION** nə olduğunun flowchartı göstərilmişdir.

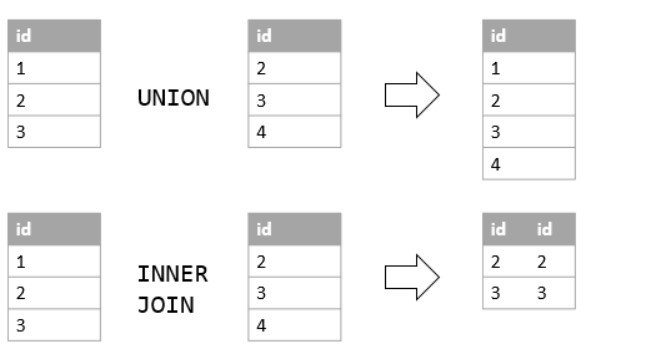
yuxarıdakı şəkildə gördüyümüz kimi **A** table-ı 1,2,3 və **B** table-ı isə 3,4,5 dəyərlərini saxlamaqdadır. Nəticə olaraq **A** və **B** table-ları **UNION** olunaraq bir **resultset** table-ında saxlanılmaqdadır, lakin fikir versəniz hər iki table-da da təkrar dəyərlər var, lakin onlar **resultset**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

**UNION ALL**

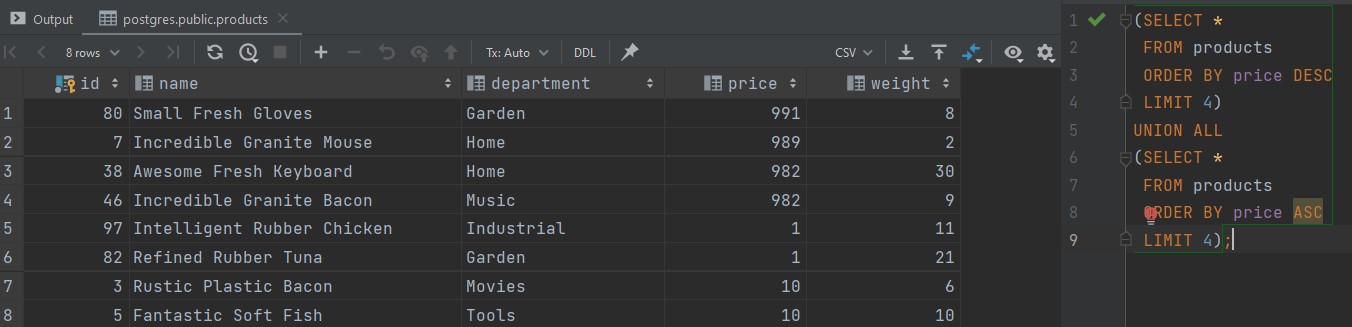
**UNION** və **UNION ALL** demək olarki eynişeydir, lakin onlar arasında performans cəhətindən önəmli fərqliliklər vardır. **UNION ALL** da **UNION** kimi iki və ya daha çox fərqli table-lardan sətirdəki məlumatları birləşdirərək onları bir **resultset** də çıxartmağımıza şərait yaradır, lakin **UNION**-dan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

**UNION vs JOIN**

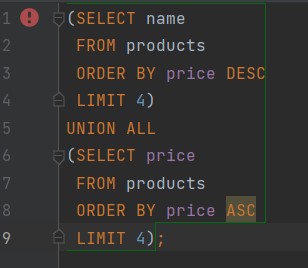
**Join**-lərdən məsələn **INNER JOIN** və yaxud **LEFT JOIN** iki table-dan column-ları birləşdirərkən, **UNION/ALL** iki querydən sətirləri birləşdirməkdədir. Başqa deyimlə desək, **join** nəticələri horizontal yəni yanaşı bitişdirərkən, **UNION** vertical yəni yuxarıdan aşağ şəklində bitişdirir nəticələri,eynən 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.

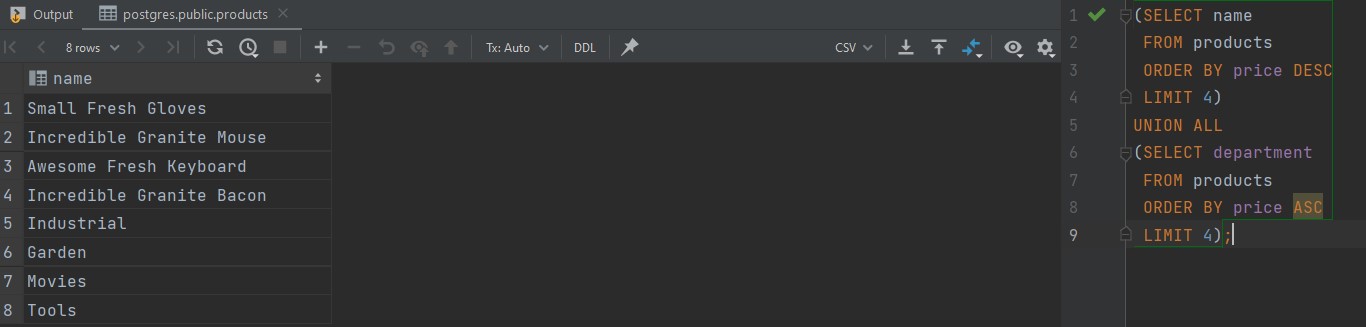
* **NOTE:** Set operatorlarından istifadə edərkən bilinməsi lazım olan bəzi məqamlar vardır.

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

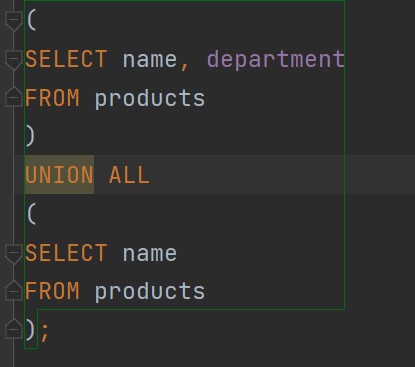




gördüyünüz kimi, yuxarıdakı query də birinci **SELECT** də **name**, ikinci **SELECT** də isə **price** sütunları kimi fərqli tipdə sütunları çəkmək istədiyimizdən xəta ilə qarşılaşdıq.

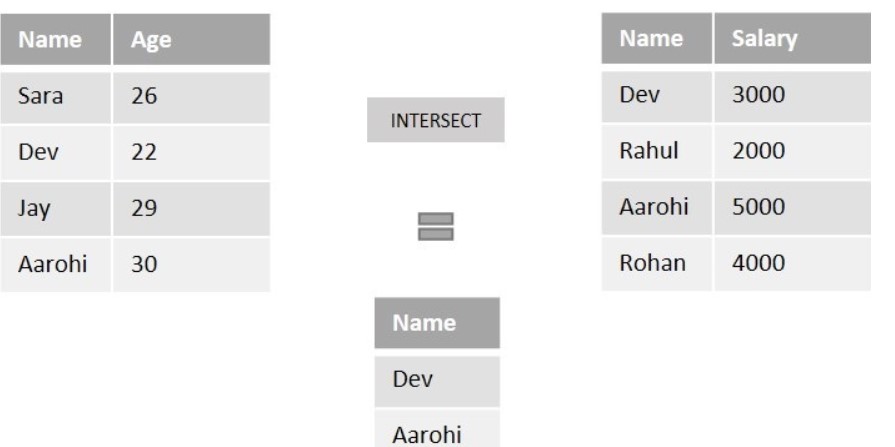
1. column type-lar eyni lakin fərqli sütunlara görə də set operatorlarından istifadə edə bilərsiniz, eynən aşağıdakı şəkildə olduğu kimi.

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.

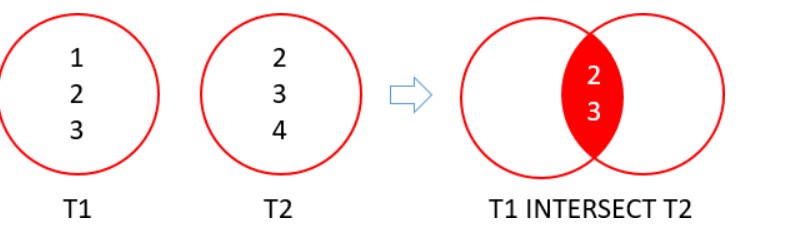
1. **UNION** uniqueness dataları gətirərkən, **UNION ALL** duplicate-ləridə gətirəcəkdir.
2. **column**-ların sayı eyni olmalıdır hər iki querydə də əks halda xəta ilə qarşılaşacayıq, eynən aşağıdakı şəkildə olduğu kimi.

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

**INTERSECT/ALL**

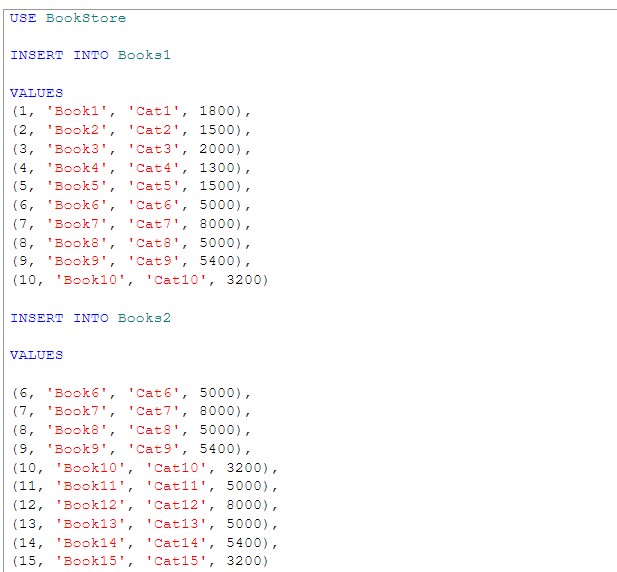
Bu **set** operatoru isə iki və ya daha çox table-ın hər ikisindədə ortax məxrəc olan sətirdəki məlumatları gətirmə işinə baxacaqdır. Əgər bir query-də olan **result**,digər query-də yoxdursa, həmən olmuyan-lar ortadan götüzdürləcəkdir, yalnızca biri-birinin qarşılığı olan sətirdəki məlumatları **resultset**-ə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

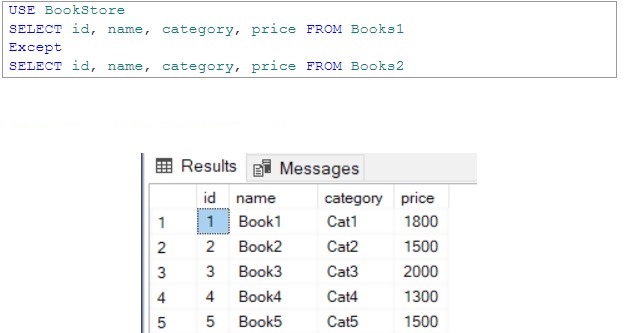
Yuxarıdakı şəkildəndə gördüyünüz kimi, **T1** və **T2** table-larında yalnızca hər iki tableda biri-birinin qarşılığı olan sətirlərdəki məlumatlar **resultset**-ə daxil olundu, digərləri isə aradan götürüldü.

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

**EXCEPT**

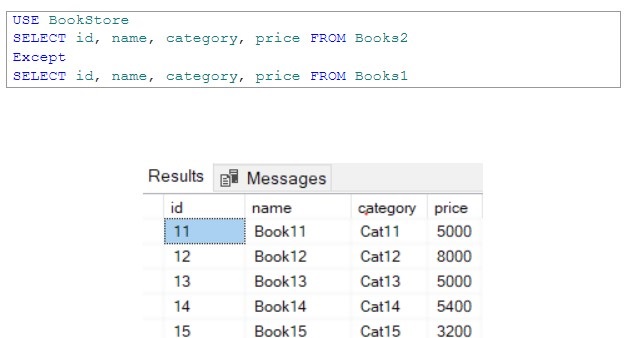
**EXCEPT** set operatoru ondan əvvəl yazılan **SELECT** statement-indən qayıdan nəticə, ondan sonra yazılan **SELECT** statement-indən qayıdan nəticədə olmazsa, həmən əvvəl yazılan **SELECT** statement-indən qayıdan nəticə **resultset**-ə daxil olunur. Bunu daha yaxşı anlamaq üçün gəlin aşağıdakı nümunələrə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

Yuxarıdakı şəkildəndə gördüyünüz kimi biz, **Books1** və **Books2** table-larına datalar **INSERT** etmiş olduq və **Books1** table-ında 1-dən 10-a qədər sətirlər olduğu halda, **Books2** table-ında isə 6-dan 15-ə qədər olan sətirlər vardır.

1. İndi isə gəlin **insert** etdiyimiz bu table-lar üzərində **EXCEPT** operaoturnu istifadəsinə baxaq.

Yuxarıdakı şəkildəndə gördüyünüz kimi bizim yazmış olduğumuz query-dən yalnızca 1-dən 5-ə qədər olan sətirdəki məlumatlar **resultset** də saxlanıldı. Bu isə onun göstəricisidir ki, az öncədə dediyimiz kimi əgər **EXCEPT** dən əvvəl yazılan **SELECT** query-sindən qayıdan nəticələr, **EXCEPT** dən sonra yazılan **SELECT** query-sindən qayıdan nəticələr arasında olmuyanları bizə ver. Table-lara insert etdiyimiz datalar arasındanda gördüyümüz kimi **Books1** table-ında 1-dən 5-ə qədər olan sətirdəki məlumatlar **Books2** table-ında olmadığından məhz biz bu dataları əldə etmiş olduq, yerdə qalan sətirdəki datalar isə ortadan qaldırıldı.

1. Az öncə yazdığımız query-nin birə-bir eynisində table-ların yerini dəyişmiş olsaq, bu nəticəni almış olarıq, eynən aşağıdakı şəkildə olduğu kimi.



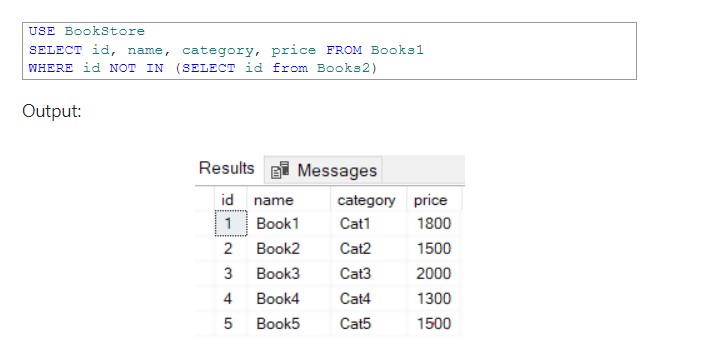
Yuxarıdakı şəkildəndə gördüyünüz kimi bu dəfə isə **EXCEPT** -in sol hissəsinə **Books2** table-ı əlavə edildi, digər **Books1** isə sağ tərəfə elan edildi, beləliklə bu dəfə isə sol tərəfdəki **SELECT**-dən qayıdan **resultset** sağ tərəfdəki **resultset**-də yoxdursa həmən olmuyanlar bizim **resultset**­-ə daxil olunacaqdır.

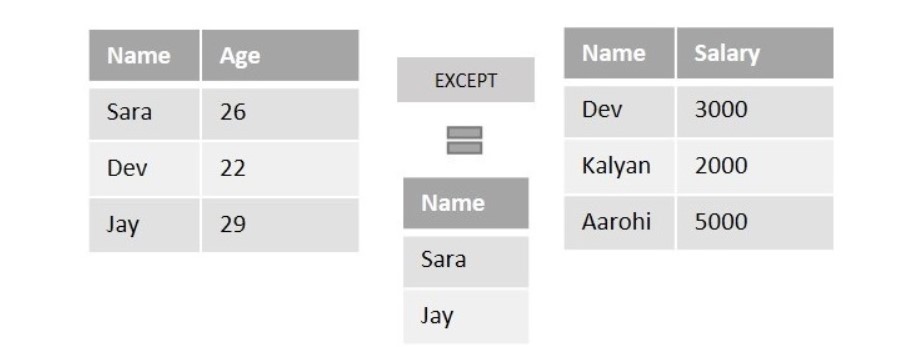
**EXCEPT vs NOT IN**

**EXCEPT** operatorunun necə işləməsini bildiyimizə görə, **SQL** də **EXCEPT** ilə **NOT IN** operatorları arasında fərqliliyi anlamağımız gərəkdir. Aşağıda bu operatorlar arasında 2 əsas fərq göstərilmişdir.

1. **EXCEPT** operatoru iki və ya daha çox **SELECT** statementdən qayıdan yalnızca ortax məxrəcə aid olan sətirləri qaytararkən, **NOT IN** duplicate-ləri də əlavə edə bilər.
2. **EXCEPT** operatorunda müqayisə əməliyyatı 2 **SELECT** statement arasında gedərkən, **NOT IN** isə müqaiysə əməliyyatını tək bir **column** dəyərinə görə aparmaqdadır.

aşağıdakı şəkildə olan nümunə **NOT IN** operatorunun **EXCEPT** operatoruna bənzər sorğusu göstərilmişdir.

gördüyünüz kimi yuxarıdakı sorğuda biz **Books1** table-ında **id** dəyəri **Books2** table-ında id dəyərinə bərabər olmuyan sətirdəki məlumatları çəkməsini istədik. Bunu daha yaxşı anlamaq üçün, belə açıqlama vermək olar ki, əgər **Books1** table-ında olan id sütununda yerləşən hər hansı bir dəyər **Books2** table-ında olan id sütununda olan dəyərə bərabər olmuyanları çəkib **Books1** table-ından çəkməsini istədik. Buradanda gördüyünüz kimi biz bunu **NOT IN** operatoru vasitəsi ilə tək bir sütuna görə müqayisə əməliyyatını aparmış olduq, lakin **EXCEPT** set operatorunda isə 2 **SELECT** statement arasında getməkdədir.

Daha sonra gəlin başqa bir simulyasiyaya baxaraq **EXCEPT**-i anlamağa çalışaq.