```
Create Database Notlandırma;
Use Notlandırma;
-- drop database Notlandırma;
Show Databases;
-- bolumler tablosu
Create Table IF NOT EXISTS bolumler(
bolno INT AUTO_INCREMENT PRIMARY KEY NOT NULL,
bolum VARCHAR(50) NOT NULL
);

Create Table IF NOT EXISTS ogrenciler(
ogrno INT AUTO_INCREMENT PRIMARY KEY NOT NULL,
tcno DECIMAL(11) UNIQUE NOT NULL,
ad VARCHAR(50) NOT NULL,
soyad VARCHAR(50) NOT NULL,
bolno INT NOT NULL,
FOREIGN KEY(bolno) REFERENCES bolumler(bolno)
);

Create Table IF NOT EXISTS dersler(
dersno INT AUTO_INCREMENT PRIMARY KEY NOT NULL,
ders VARCHAR(50) NOT NULL,
kredi INT DEFAULT '4' NOT NULL
);
Create Table IF NOT EXISTS notlar(
notid INT AUTO_INCREMENT PRIMARY KEY NOT NULL,
ogrno INT NOT NULL,
dersno INT NOT NULL,
```

```
FOREIGN KEY(ogrno) REFERENCES ogrenciler(ogrno),
FOREIGN KEY(dersno) REFERENCES dersler(dersno),
vize float NOT NULL,
final float NOT NULL
);

Create Table IF NOT EXISTS egitmenler(
egitmenno INT AUTO_INCREMENT PRIMARY KEY NOT NULL,
ad VARCHAR(50) NOT NULL
);
-- Ekleme
Insert Into bolumler(bolum)
Values ('Elektrik'),
('Bilgisayar'),
('Mekatronik'),
('Peyzaj'),
('Makine'),
('Tıp')
;
Select * from bolumler;
Insert Into bolumler(bolum)
Values ('Fizik'),
('Kimya');
Select * from bolumler;
Insert Into ogrenciler(tcno,ad,soyad,bolno)
Values ('2451','Ali','Kaya',2),
('2452','Ahmet','Kar',2),
('2253','Mehmet','Demir',1),
```

('2454','Yusuf','Duman',3),

('2152','Ahmet','Er',5),

('2652','Yasin','Han',3);

Select * from ogrenciler;

Insert Into dersler(ders,kredi)

Values ('Algoritma','4'),

('Veri Yapıları','5'),

('Matematik','4'),

('Görüntü İşleme','3'),

('Nesne Tabanlı','5'),

('Veri Madenciliği','4'),

('Ağlar','3'),

('Akışkanlar','4');

Select * from dersler;

Insert Into notlar(ogrno,dersno,vize,final)

Values ('1','2','50','60'),

('1','4','82','96.5'),

('2','2','95','49'),

('2','5','61','60'),

('3','6','13.7','85'),

('4','5','50','69'),

('4','7','49','47'),

('4','4','75','72.6'),

('5','2','68','60'),

('5','1','53.5','28');

Select * from notlar;

Insert Into egitmenler(ad)

Values ('Ali'),

('Yusuf'),

('Mehmet'),

('Yasin');

select * from egitmenler;

SORGULAR

-- join sorguları

-- join: Birden fazla tablodaki satırları, ortak bir sütun üzerinden birleştirerek sonuç kümesini genişletir.

-- Sütun birleştirme: Inner Join, Outer Join (Left, Right, Full)

-- Satır Birleştirme: Union, Intersect, Except

-- product

select * from ogrenciler, bolumler;

-- theta join

select * from ogrenciler, bolumler where ogrenciler.bolno=bolumler.bolno;

-- 1. Inner Join: İki tablonun yalnızca ortak bir sütun üzerinden eşleşen satırlarını döner.

-- Tüm öğrencilerin ad, soyad ve hangi bölümde olduklarını listeleyin:

select ogrenciler.ad, ogrenciler.soyad, bolumler.bolum from ogrenciler INNER JOIN bolumler on ogrenciler.bolno=bolumler.bolno;

-- Öğrencilerin aldıkları dersleri ve notları listeleyin:

select * from ogrenciler;

select * from dersler;

select * from notlar;

select * from ogrenciler INNER JOIN notlar on ogrenciler.ogrno=notlar.ogrno

```
INNER JOIN dersler on notlar.dersno=dersler.dersno;



-- vize notu 50 den küçük olan öğrenciler

select * from ogrenciler INNER JOIN notlar on ogrenciler.ogrno=notlar.ogrno

INNER JOIN dersler on notlar.dersno=dersler.dersno where notlar.vize<50;

-- final notu 70 den büyük olan öğrencilerin final notuna +10 ekleme

select ogrenciler.ad, ogrenciler.soyad, notlar.vize, notlar.final, (notlar.final+10) as toplam
from ogrenciler INNER JOIN notlar on ogrenciler.ogrno=notlar.ogrno

INNER JOIN dersler on notlar.dersno=dersler.dersno where notlar.vize>70;

-- vize ve final notları toplamı 90 den küçük olan öğrenciler

select ogrenciler.ad, ogrenciler.soyad, notlar.vize, notlar.final, (notlar.vize+notlar.final) as
toplam from ogrenciler INNER JOIN notlar on ogrenciler.ogrno=notlar.ogrno

INNER JOIN dersler on notlar.dersno=dersler.dersno where (notlar.vize+notlar.final)<90;

-- Vizenin %40'ının ve finalin %60'ının toplamı 50'nin altında olan öğrenciler

select ogrenciler.ad, ogrenciler.soyad, notlar.vize, notlar.final,
(notlar.vize*0.4+notlar.final*0.6) as toplam from ogrenciler INNER JOIN notlar on
ogrenciler.ogrno=notlar.ogrno

INNER JOIN dersler on notlar.dersno=dersler.dersno where
(notlar.vize*0.4+notlar.final*0.6)<50;

-- Her bölümdeki öğrencilerin aldıkları dersleri ve notları listeleyin:

select * from ogrenciler INNER JOIN notlar on ogrenciler.ogrno=notlar.ogrno

INNER JOIN dersler on notlar.dersno=dersler.dersno
```

```
INNER JOIN bolumler on ogrenciler.bolno=bolumler.bolno;



-- Bilgisayar bölümündeki öğrencilerin aldıkları dersleri ve notları listeleyin:

select * from ogrenciler

INNER JOIN notlar on ogrenciler.ogrno=notlar.ogrno

INNER JOIN dersler on notlar.dersno=dersler.dersno

INNER JOIN bolumler on ogrenciler.bolno=bolumler.bolno where
bolumler.bolum="bilgisayar";


-- Her ders için en yüksek final notunu alan öğrenciyi bulun:

select ogrenciler.ad, ogrenciler.soyad, notlar.final, dersler.ders from ogrenciler

INNER JOIN notlar on ogrenciler.ogrno=notlar.ogrno

INNER JOIN dersler on notlar.dersno=dersler.dersno

where notlar.final=(select max(n.final) from notlar as n where n.dersno=notlar.dersno);


select ogrenciler.ad, ogrenciler.soyad, notlar.final, dersler.ders from ogrenciler

INNER JOIN notlar on ogrenciler.ogrno=notlar.ogrno

INNER JOIN dersler on notlar.dersno=dersler.dersno

where notlar.final=(select max(n.final) from notlar as n);



-- 2. Left join

-- Tüm öğrencileri ve varsa aldıkları dersleri listeleyin. Ders kaydı olmayan öğrenciler de
görünsün:

select * from ogrenciler

LEFT JOIN notlar on ogrenciler.ogrno=notlar.ogrno

LEFT JOIN dersler on notlar.dersno=dersler.dersno;
```

-- 3. Right Join

-- Tüm öğrencileri ve aldıkları dersleri listeleyin. Alınmayan dersler de varsa görünsün:

```
select * from ogrenciler
RIGHT JOIN notlar on ogrenciler.ogrno=notlar.ogrno
RIGHT JOIN dersler on notlar.dersno=dersler.dersno;
```

-- 4. Full Outer Join

-- Tüm öğrencileri ve aldıkları dersleri listeleyin. Ders kaydı olmayan öğrenciler de, alınmayan dersler de görünsün:

```
select * from ogrenciler
LEFT JOIN notlar on ogrenciler.ogrno=notlar.ogrno
LEFT JOIN dersler on notlar.dersno=dersler.dersno
UNION
select * from ogrenciler
RIGHT JOIN notlar on ogrenciler.ogrno=notlar.ogrno
RIGHT JOIN dersler on notlar.dersno=dersler.dersno;
```

-- 5. Cross Join

-- Tüm öğrencilerin tüm bölümler ile olası kombinasyonlarını listeleyin:

```
select * from ogrenciler CROSS JOIN bolumler;
```

-- 6. UNION

-- ogrenciler ve egitmenler tablolarındaki kişilerin isimlerini tek bir liste halinde tekrarsız göstermek:

```
select * from ogrenciler;
select * from egitmenler;
select ogrenciler.ad from ogrenciler UNION select egitmenler.ad from egitmenler;
```

-- 7. INTERSECT (Kesişim)

-- MySQL doğrudan INTERSECT desteği sunmaz, ancak INNER JOIN ile aynı sonuç elde edilebilir.

-- ogrenciler ve egitmenler tablolarında ortak adı olan kişileri listelemek istiyoruz.

select * from ogrenciler INNER JOIN egitmenler on egitmenler.ad=ogrenciler.ad;

-- 8. EXCEPT (Fark)

-- MySQL doğrudan EXCEPT desteği sunmaz, ancak LEFT JOIN ve WHERE kullanılarak aynı sonuç elde edilebilir.

-- ogrenciler tablosunda olup, egitmenler tablosunda olmayan kişilerin adlarını listelemek:

select * from ogrenciler LEFT JOIN egitmenler on ogrenciler.ad=egitmenler.ad where egitmenler.ad IS NULL;