

3. der → son

→ VTYS - 2. hafta

1. hafta → SQL uygulamaları (Veritabanına Hızlı Giriş)

2. hafta → İlişkisel Cebir (Relational Algebra) (Tablolar cebiri)

Relation : tablo

relationship : 2 varlık arasındaki ilişki

Veritabanında 2 şey sokabilir
→ varlıklar
→ varlıklar arasındaki ilişki } sokabilir

İlişkiler 2 varlığa ihtiyacı vardır (en az 2)

Veritabanında ne sokabilir?

1) Varlık türü 2) İlişkiler türleri → Tablo

Tablo = kayıt türü (Nüfus ya da varlığı gösterir)

Veritabanı = Tablo türü

varlık ve ilişki türlerini belirlemek tablo belirlemek → Veritabanı türü
varlıklar belirledikten sonra ilişkileri de belirlemeliyiz!
örneğin → varlık türü → ilişki türü → no. adisayısı
örneğin → primary key (en az bir ilişki türü)

Veritabanı türü! (İstisnalar olabilir ama genelde böyle)

VTYS → program (veri okumayı, yazmayı sağlar)

Veri türleri → Veritabanı türünde yapılar komutları (2 tür komut var)
→ tablo türü komutları → tablo yapıları komutları
insert, delete, update, select → create, drop, alter
write (değiştirme) → read (okuma) → oluşturma, silme, değiştirme
modification → retrieve

SQL (Veritabanı sorgu dili → Yaratılabilir sorgu dili)

Query → Veritabanında bilgiye erişim şekli

procedural → adimsal (farklı) dil

Kayıt seçerken bir kayıta birimle çalışır. Veritabanı türünden

Kayıt Seçme Dilemi

Sorgu: Bize mch. bölümündeki öğrencilerin kayıtlarını listele

$\pi_{bno=b}$ (öğrenci)

SELECT * FROM öğrenci WHERE bno=b

❖ Duplicate elimination → aynı kayıtları 1 kere yazılması

* Sorgu: dersol tablosundaki derslerin kodlarını listele

proje
op π_{kod} (dersol)

kodu
ce 101
ce 301

select distinct kodu
from dersol

tablo oluşturulmuş
sadece kodlarını gösteriyor.

duplicate elimination yapar

create table derskodun as select distinct kodu from dersol
↳ bu şekilde yavaş seçtiklerini tablo haline getirebilirsin.

* Sorgu: bpr ve hoca isimlerinin listelerini göster.

$\pi_{adi}(\sigma_{prci}) \cup \pi_{adi}(\sigma_{hoca}) \rightarrow$ Sayı tabanlı sonuçlar
6 kay. 2 dir

$(select adi from öğrenci) \cup (select adi from hoca);$

* Sorgu: öğrenci olarak kullanılan fakat hoca olarak
kullanılmayan isimleri listele

$\pi_{adi}(\sigma_{prci}) - \pi_{adi}(\sigma_{hoca})$

$(select adi from öğrenci) \text{ minus } (select adi from hoca)$

SQL union / intersect / minus otomatik olarak dup. elimi. yapar.

$\rho_X(E) \rightarrow$ yeniden isimlendirme
E yerine X demek BTPR Renomme

$\sigma_{prci}(\sigma_{prci}) \cup öğrenci$

$\pi_{sno}(\sigma_{prci}(sno, fno, lno, dno))(\sigma_{prci})$

1. Soru: 'A' den öğrencilerin numaralarını bulmak

$\Pi_{opno} (F_{not=A}(dersol))$

select opno from dersol where not = 'A'

2. Soru: Not A olan öğrencilerin adlarını bulmak

$\Pi_{ad} (dersol \text{ opno } (dersol) \text{ gr. opno } \wedge \text{ not } = 'A' \text{ koşulu olmali})$	1. Adı	2. Adı	3. Adı
	1. Adı	1. Adı	1. Adı
	1. Adı	1. Adı	1. Adı
	2. Adı	2. Adı	2. Adı
	2. Adı	2. Adı	2. Adı

3. Ne zaman bir karteğin yansıması bir koşul yansıması kullanılır? latın

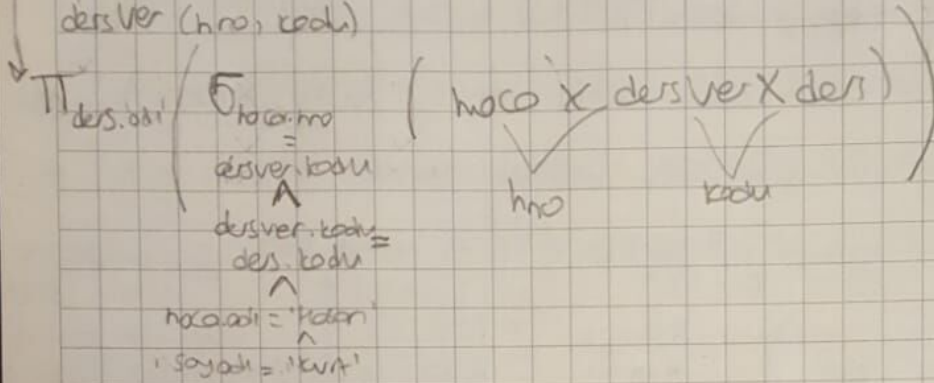
Bir tablonun primary key'i başka bir tabloda kullanırsa foreign key olur. (yabancı anahtar)
 öğrenci tablosundaki opno = primary key ama dersAl tablosunda opno foreign key olur.
 Primary key tabloda 1 kez kullanılır. (Anahtarsız tablo olmaz) !!
 dersAl tablosunda primary key opno ve kodu birlikte!
 Bir öğrenci 1 ders 2 kere okutabilir çünkü primary key 1 kez yazılır.

3. select ad
 1. from dersol, öğrenci
 2. where dersol.opno = öğrenci.opno and dersol.not = 'A'

1. dersAl X öğrenci → t (ortama girer)
 2. $\sigma(t) \rightarrow k$
 3. $\Pi(k)$

*Soru: Hoca Kurt adı hocaın verdiği dersin imkını İstek

hoca (hno, adi, soyadi, bno)
ders (kod, adi, kredisi, hno)
dersver (hno, kod)



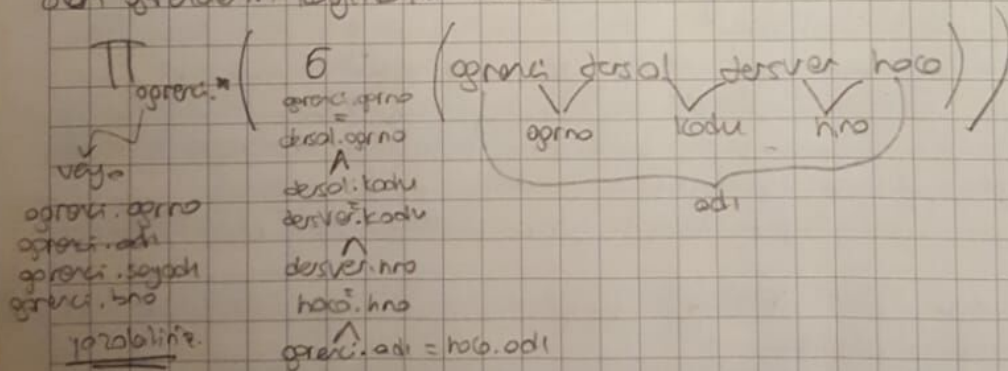
select d.adi

from hoca h, dersver dv, ders d

where h.hno=dv.hno and dv.kodu=d.kodu

and h.adi='Kurt' and soyadi='Kurt'

*Soru: İmri hocasıyla aynı olup da onun verdiği derslerin (bir den) olan öğrencilerin kayıtlarını İstek



VTYS

Relational Algebra + SQL (Yabancı) (tek print) MySQL

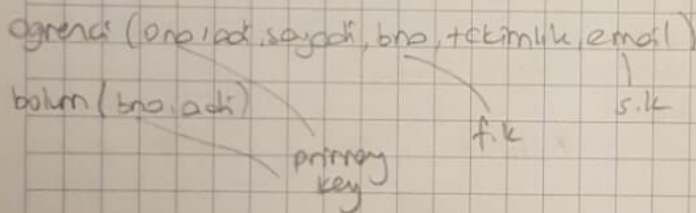
Sorguyu nasıl ifade ederiz?

1. Tabloları belirle
2. "A" operatörlerini belirle

$$\bigcup_{t_1, a_1}^{t_2, a_2} (t_1(a_1 \dots) \times t_2(a_2 \dots a_n))$$

foreign key (yabancı)
primary key (birinci)

Tabloda ikinci anahtar yani secondary key // candidate (aday)



→ p.k olanı null olamaz ama sk ve f.k null olabilir.

insert into ogrenci (null, 'Ali', 'Kurt', 15, null, null)

error

öğrenci eklemek için öğrenci numarası olması gerekir

insert into ogrenci (2001, 'Ali', 'Kurt', 20, null, null)

delete from bolum where bno = 20

bno	adi
10	EE
20	IE
30	CE

create table ogrenci (ono int primary key, adi char(10), soyad char(10), bno int foreign key referencing (bolum) on delete cascade, email char(10);

❌ bölüm tablosundan foreign key alırsanız hata olur

❌ intersection yapabilmek için sütun sayısı ve tipi aynı olmak zorundadır. İsimler aynı olması da olur.

Soru 1: Öğrencilerin isimleri ve bu öğrencilerin aldığı derslerin isimlerini listele

ali, engin
veli, data science
veli, vbys

$\Pi_{\text{öğrenci.no, ders.no}} \left(\sigma_{\text{öğrenci.no} = \text{ders.ol.no}} (\text{öğrenci} \times \text{ders.ol} \times \text{ders}) \right)$

öğrenci.no, öğrenci.no, ders.no, ders.ol.no, ders.ol.kodu, ders.kodu

$\Pi_{\text{öğrenci.no, ders.ol.no}} \left(\sigma_{\text{öğrenci.no} = \text{ders.ol.no}} (\text{öğrenci} \times \text{ders.ol} \times P(\text{ders})) \right)$

adı, ders adı, ders (kod, ders adı, kredi, ders.no)

Soru 2: Öğrenci no ve bölümün adını listele

$\Pi_{\text{öğrenci.no, b.adı}} \left(\sigma_{\text{öğrenci.no} = \text{bölüm.no}} (\text{öğrenci} \times \text{bölüm}) \right) = \Pi_{\text{öğrenci.no, b.adı}} (\text{öğrenci} \times \text{bölüm})$

select ö.no, b.adı
from öğrenci o, bölüm b
where o.no = b.no

Soru 3: Aldığı dersin hocasıyla adı aynı olan öğrencilerin adını listele

$\Pi_{\text{öğrenci.no, ders.ol.no, ders.ver.no}} \left(\sigma_{\text{öğrenci.no} = \text{ders.ol.no}} (\text{öğrenci} \times \text{ders.ol} \times \text{ders.ver}) \right)$

öğrenci.no, ders.ol.no, ders.ver.no, ders.ol.kodu, ders.ver.kodu, ders.hoca.no, ders.hoca.adı

$\Pi_{\text{öğrenci.adı}} \left(\sigma_{\text{öğrenci.no} = \text{ders.ol.no} \wedge \text{ders.ol.kodu} = \text{ders.ver.kodu} \wedge \text{ders.hoca.no} = \text{öğrenci.no}} (\text{öğrenci} \times \text{ders.ol} \times \text{ders.ver} \times \text{hoca}) \right)$

öğrenci.no, ders.ol.no, ders.ol.kodu, ders.ver.kodu, ders.hoca.no, ders.hoca.adı

select öğrenci.adı
from öğrenci o, ders ol, ders ver, hoca h
where o.no = ol.no and ol.kodu = ver.kodu and
ol.hoca = h.no and o.adı = h.adı

$r \div s$

$R = (A_1, \dots, A_m, B_1, \dots, B_n)$
 $S = (B_1, \dots, B_n)$

$\left. \begin{array}{l} R' \text{ y} S' \text{ ye bağımlı olan tüm} \\ S' \text{ in tüm elemanlarının } R' \text{ de} \\ \text{olması gerekir} \end{array} \right\}$

Soru: 'Database' adlı veri den sorguların adlarını listele

$\Pi_{\text{sorgular}} \left(\sigma_{\text{sorgular.ano} = \text{dersol.ano} \text{ and } \text{dersol.kodu} = \text{ders.kodu}} (\text{sorgular} \times \text{dersol} \times \text{ders}) \right)$

$\text{dersol.kodu} = \text{ders.kodu}$
 $\text{ders.ad} = \text{"Database"}$

select s.ad
 from sorgular s, dersol ds, ders d
 where s.ano = ds.ano and ds.kodu = d.kodu and d.ad = 'Database'

Soru: Tüm derslerin sorguların adlarını listele

$y(\text{ano}, \text{kodu}) \div x(\text{kodu})$

$\Pi_{\text{ano, kodu}}(\text{ders}) \div \Pi_{\text{kodu}}(\text{ders})$

$\Pi_{\text{ad}}(\text{temp}(\text{ano}) \text{ or } \text{sorgular})$

select ano
 from sorgular s
 where not exists (
 (select kodu from ders) minus (
 select kodu from dersol
 where dersol.ano = s.ano
)
)

select ad || ' ' || soyadı from ogrenci

select concat(ad, ' ', soyadı) from ogrenci

$\Pi_{\text{ad} || \text{' ' } || \text{soyadi}}(\text{ogrenci})$

branch name | (account)

branch name | account number | balance

— account no.

branch name'e göre

grubla, bakiye
toplamlarıdır

Soru: Bilg. mch. adı bslmndeki tüm dersler olan öp.lerin no'larını listele

{oro. kodu} \div {bilg. mch. tüm derslerin kodları}

$\Pi_{oro, kodu} (dersol) \div \Pi_{kod} (ders \bowtie \sigma_{kod='Bilg. mch'} (bolum))$

select
from oprenci o
where not exists (select kodu
from ders d, bslm b
where d.bro = b.bro and
b.ad = 'Bilg. mch') minus (select kodu
from dersol
where dersol.oro = o.oro)

Soru: (her bölüm için) Bslmndeki numaraları ve öp. sayılarını listele

oro, $\Pi_{count(oro)} (oprenci)$

Soru: Bilg. mch. bslmndeki derslerin her notu kaç kişinin aldığı

$\Pi_{kod, notu, \sigma_{kod='Bilg. mch'}} (bolum) \bowtie ders \bowtie dersol$

$t_1 \leftarrow$
+ tabloya
kaydeden

$\Pi_{kod, notu} \rho_{count(oro)} (t_1)$

select + kodu, notu, count(oro)
from bolum b, ders d, dersol do
where b.bro = d.bro and b.kodu = d.kodu and b.ad = 'Bilg. mch'
group by kodu, notu

kodu	notu	count(oro)
ce101	A	2
ce101	B	5
ce101	C	10
ce101	D	1
ce201	A	10
ce201	B	10
ce201	F	20

Sorgu: her bölimdeki öğrenci sayılarını listele

select bno, count(ono) ← *select'te sadece group by
from öğrenci
where bno ve attribute sınırları ya da
agg func kullanılabilir

(sayısal olduğunu kabul et)

her ders için öğrenci sayısı ve not ortalaması

Kodu g (dersid)
count(ono), avg(notu)

select kodu, count(ono) as sayisi, avg(notu) as ort
from dersol
group by kodu

Kodu	sayisi	ort
bel01	55	92.7
ce201	102	25.0