3 dev → sah

— VTYS — 2.hafta

1. hafta → SQL uygulaması (Veritabanına Hızlı Giriş)

2. hafta → İlişkisel Cebir (Relational Algebra) (Tablosal cebir)

Relation : tablo
relationship : 2 varlık arasındaki ilişki

Veritabanında 2 şey saklanır → varlıklar
→ varlıklar arasındaki ilişkiler } saklanır

İlişki için 2 varlığa ihtiyaç vardır (en az 2)

Veritabanında ne saklanır?
1. varlık kümesi 2) ilişki kümeleri → Tablo

Tablo = kayıt kümesi (ilişkiyi ya da varlığı gösterir)

Veritabanı = Tablo kümesi

varlık ve ilişki kümelerini belirleyip tabloları oluşturmak → veritabanı tasarımı
varlıkları belirledikten sonra özelliklerini de belirlemeliyiz!
öğrenci → varlık kümesi özellikleri → no, ad, soyad
ogrNo → primary key (eşsiz bir özelliktir)

Veri tablosu dolmaz! (üst sınırı olabilir ama genelde dolmaz)

VTYS → program (veriyi okumayı, yazmayı sağlar)

Veri üzerinde, tablo üzerinde yapılan komutlar (2 tür komut var) → 7 tane
tablo verisi komutlar | → tablo yapısı komutları
insert, delete, update, select | create, drop, alter
write (değiştiriyor) modification | read (değiştirmiyor) retrieve | oluştur, sil, değiştir

SQL (Veri tabanı sorgu dili — Yapısal sorgu dili) Structured Query Lang.

Query → veri tabanında bilgiye erişim isteği

procedural → adımsal (tanımsal) dil

kayıt seçerken tüm kayıtları kontrol etmek zorundayız
(kayıt seçme işlemi yaparken)

Kayıt seçme işlemi

Sorgu: Bilgisayar müh. Bölümündeki öğrencilerin kayıtlarını listele

$\sigma_{bno=6}$ (ogrenci)

SELECT * FROM ogrenci WHERE bno=6

* duplicate elimination → aynı kayıtların 1 kere yazılması

* Sorgu: dersal tablosundaki derslerin kodlarını listele

project op: Π_{kodu} (dersal)

kodu
ce 101
ce 301

select DISTINCT kodu from dersal → duplicate elimination yapar

tablo oluşturmuyor sadece istediklerini gösteriyor.

create table derskodları as select distinct kodu from dersal
↳ bu şekilde yaparak seçtiklerini tablo haline getirebilirsin.

* Sorgu: öğr ve hoca isimlerinin listesini göster

$\Pi_{adı}$ (ogrenci) $\cup$ $\Pi_{adı}$ (hoca) → slayta göre sonuçta 6 kayıt olur

(select adı from ogrenci) union (select adı from hoca);

* Sorgu: öğr ismi olarak kullanılan fakat hoca ismi olarak kullanılmayan isimleri listele

$\Pi_{adı}$ (ogrenci) $-$ $\Pi_{adı}$ (hoca)

(select adı from ogrenci) minus (select adı from hoca)

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

$\rho_x(E)$ → yeniden isimlendirme, E yerine x demek istiyor — Rename

$\rho_{student}$(ogrenci) $\cup$ ogrenci

Π_{sno} ($\rho_{student}$ (sno, fname, lname, dno))(ogrenci);

(*) Sorgu: "A" notu alan öğrencilerin numaralarını listele

$$\pi_{ogrno}\left(\sigma_{not='A'}(dersal)\right)$$

select ogrno from dersal where not = 'A'

→ * Sorgu: Notu A olan öğrencilerin adlarını listele

(dersal X ogrenci) → ~~1 ali kurt → 2 ce101 B~~

1 ali - - -	1 ce101 A
1 ali - - -	1 ce301 A
2 Ayşe - - -	2 ce101 B
~~2~~	~~1 ce101 A~~
2	
~~3 Ayşe Demir~~	~~2 ce101 B~~
~~3~~	1
3	1

Bir tablo olmaz çünkü kayıtlar karıştı sadece onların eşleşmeleri kullanmanız lazım → koşul yaz!

$\pi_{ad}\left(\sigma_{dersal.ogrno = ogr.ogrno \wedge not='A'}(dersal)\right)$ koşulu olmalı

(*) Ne zaman bir kartezyen yaparsak bir koşul yani kayıt seçme operatörü kullanmanız lazım!

Bir tablonun primary key'i başka bir tabloda kullanılırsa foreign key dir (yabancı anahtar)

öğrenci tablosundaki ogrNo → primary key ama dersAl tablosunda ogrNo foreign key olur

Primary key tabloda 1 kez kullanılır. (Anahtarsız tablo olamaz)!

dersAl tablosunda primary key ogrNo ve kodu birlikte!

Bir öğrenci 1 dersi 2 kere alamaz çünkü primary key 1 kez yazılır.

3 select adı
1 from dersal, ogrenci
2 where dersal.ogrNo = ogrenci.ogrNo and dersal.not = 'A'

(ortaya çıkış sırası)

1 dersal X ogrenci → t
2 $\sigma(t) \rightarrow k$
3 $\pi(k)$

GIPTA

*Sorgu: Hasan kurt adlı hocanın verdiği derslerin dersin bilgisini listele

hoca (hno, adı, soyadı, bno)
ders (kodu, adı, kredisi, hno)
dersVer (hno, kodu)

$\Pi_{ders.adı}\left(\sigma_{\substack{hoca.hno = dersVer.kodu \\ \wedge\ dersVer.kodu = ders.kodu \\ \wedge\ hoca.adı = 'Hasan' \\ \wedge\ soyadı = 'Kurt'}}\left(hoca \times dersVer \times ders\right)\right)$

hno, kodu

```
select d.adı
from hoca h, dersVer dv, ders d
where h.hno=dv.hno and dv.kodu=d.kodu
      and h.adı='Hasan' and soyadı='Kurt'
```

*Sorgu: ismi hocasıyla aynı olup da onun verdiği dersten (bir ders) olan öğrencilerin kayıtlarını listele

$\Pi_{ogrenci.*}\left(\sigma_{\substack{ogrenci.ogrno = dersal.ogrno \\ \wedge\ dersal.kodu = dersVer.kodu \\ \wedge\ dersVer.hno = hoca.hno \\ \wedge\ ogrenci.adı = hoca.adı}}\left(ogrenci \times dersal \times dersVer \times hoca\right)\right)$

ogrno, kodu, hno

adı

veya
ogrenci.ogrno
ogrenci.adı
ogrenci.soyadı
ogrenci.bno

yazılabilir.

VTYS

Relational Algebra + SQL
(yaparak) (text print)
MySQL

Sorguyu nasıl ifade ederiz?

1. Tabloları belirle
2. RA operatörlerini belirle

$\sigma_{t_1.a_1 = t_2.a_1} (t_1(a_1 \ldots) \times t_2(a_2 \ldots a_1))$

primary key (birincil) — foreign key (yabancı)

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

ogrenci (ono, ad, soyadi, bno, tckimlik, email)
bolum (bno, adi)

primary key — f.k — s.k

→ p.k alanı null olamaz ama sk ve fk null olabilir.

insert into ogrenci(~~null~~, 'Ali', 'Kurt', 15, null, null)
→ error
Öğrenci ekleyebilmek 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), soyadi char(10), bno int foreign key
referencing (bolum) on delete cascade,
email char(10));

(*) bölüm tablosundan foreign key ol silersen bir şeyi ...

(*) intersection yapabilmek için sütun sayısı ve tipi aynı olmalı
kolonları içindekiler aynı olmasa da olur.

GIPTA

Sorgu: öğrencilerin isimleri ve bu öğrencilerin aldıkları derslerin isimlerini listele

ali, prog. giriş
veli, data science
veli, vtys

$\Pi_{ogrenci.adi,\ ders.adi} \left(\sigma_{ogrenci.ono = dersal.ono \ \wedge \ dersal.kodu = ders.kodu} (ogrenci \times dersal \times ders) \right)$

ono   kodu

natural join → ismin değiştirme

$\Pi_{adi,\ dersadi} \left(ogrenci \bowtie dersal \bowtie \rho_{ders(kodu, dersadi, kredisi, dersono)} (ders) \right)$

Sorgu: öğrenci no ve bölümün adını listele

$\Pi_{ono,\ bAdi} \left(\sigma_{ogrenci.bno = bolum.bno} (ogrenci \bowtie bolum) \right) = \Pi_{ono,\ b.Adi} (ogrenci \bowtie bolum)$

```
select o.ono, b.badi
from ogrenci o, bolum b
where o.bno = b.bno
```

Sorgu: Aldığı dersin hocasıyla adı aynı olan öğrencilerin adını listele

$ogrenci \bowtie dersal \bowtie dersver \bowtie \rho_{X(hno, adi, hsoyadi, kredi, hbno)} (hoca)$

ono   kodu   hno   adi

$\Pi_{ogrenci.adi} \left(\sigma_{ogrenci.ono = dersal.ono \ \wedge \ dersal.kodu = dersver.kodu \ \wedge \ dersver.hno = hoca.hno \ \wedge \ ogrenci.adi = hoca.adi} (ogrenci \times dersal \times dersver \times hoca) \right)$

```
select ogrenci.adi
from ogrenci o, dersal da, dersver dv, hoca h
where o.ono=da.ono and da.kodu = dv.kodu and
      dv.hno=h.hno and o.adi=h.adi ;
```

GIPTA

R ÷ S

R = (A1, ..., Am, B1, ..., Bn)
S = (B1, ..., Bn)
} R'yi S'ye bölebilmemiz için S'nin tüm alanlarının R'de olması gerekir

Sorgu: 'Database' adlı dersi alan öğrencilerin adlarını listele

$\Pi_{ogrenciadi}\left(\sigma_{\substack{ogrenci.ono = desal.ono \\ desal.kodu = ders.kodu \\ ders.adi = "Database"}}(ogrenci \times dersal \times ders)\right)$

ono, kodu

```
select o.adi
from ogrenci o, desal da, ders d
where o.ono=da.ono and da.kodu=d.kodu and d.adi = 'Database';
```

Sorgu: Tüm dersleri alan öğrencilerin adlarını listele

y (ono, kodu) ÷ x(kodu)

$\Pi_{ono, kodu}(desal) \div \Pi_{kodu}(ders)$

$\Pi_{adi}(temp(ono) \bowtie ogrenci)$

```
select ono
from ogrenci o
where not exists ((select kodu from ders) minus (select kodu from desal where desal.ono = o.ono))
```

```
select adi||' '||soyadi from ogrenci
select concat (adi, ' ', soyadi) from ogrenci
```

$\Pi_{adi||' '||soyadi}(ogrenci)$

branch_name $\gamma_{sum(balance)}$ (account) → account'ları branch_name'e göre grupla, balance'ların toplamını bul

branch_name | account_number | balance

Sorgu: Bilg. müh. adlı bölümdeki tüm dersleri alan öğrencilerin no'larını listele

$$\{ono, kodu\} \div \{\text{bilg. müh. tüm derslerin kodları}\}$$

$$\Pi_{ono, kodu}(dersal) \div \Pi_{kodu}\left(ders \bowtie \sigma_{bodi='Bilg.müh'}(bolum)\right)$$

```
select
from ogrenci o
where not exist ((select kodu
                  from dersi d, bolum b
                  where d.bno = b.bno
                  and bodi = 'Bilg. müh')
                 minus
                 (select kodu
                  from dersal
                  where derselono = o.ono))
```

Sorgu: (her bölüm için)
Bölümlerin numaralarını ve öğrenci sayılarını listele

$$bno\, \gamma_{count(ono)}(ogrenci)$$

Sorgu: Bilg. müh bölümündeki derslerin her notu kaç kişinin aldığını bulunuz

$t_1 \leftarrow \Pi_{kodu, notu, ono}\left(\sigma_{bodi='Bilg.müh'}(bolum) \bowtie ders \bowtie dersal\right)$

t_1 tablosuna kaydet

$$kodu, notu\, \gamma_{count(ono)}(t_1)$$

```
select kodu, notu, count(ono)
from bolum b, dersi d, dersal da
where b.bno = d.dno and d.kodu = da.kodu and bbodi = 'Bilg.müh'
group by kodu, notu
```

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

GIPTA

Sorgu: her bölümdeki öğrenci sayılarını listele

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

(sayısal olduğunu kabul et)

her dersi alan öğrenci sayısı ve not ortalaması

$$\text{Kodu}\ \mathcal{G}_{\text{count(ono)}, \text{avg(notu)}}(\text{dersal})$$

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

kodu	sayısı	ort
bel01	55	92.7
ce201	102	25.3