VERİ TABANI YÖNETİM SİSTEMLERİ

DÖNEM ÖDEVİ

SOSYAL MEDYA

Öğrenci Adı Soyadı : Merve BALCI
Öğrenci Numarası : 1821221022
Ödev Teslim Tarihi : 13.01.2022

Proje konumda sosyal medya, veri tabanı modelinde 16 tane tablo oluşturdum, oluşrueduğum tablolar arasında farklı ilişkiler kurdum.

1- ANALİZ BELGESİ

Kullanıcı adı

Bir kullanıcı adını aynı anda tek bir kullanıcı alabilir. Ancak zaman içinde hesap silininceb başka kullanıcılara verilebilir.

Kullanıcı

Kulanıcıcnın ad, soyad, kullanıcı adı, mail, kayıt tarihi vs gibi bilgileri vardır

Bir kullanıcının birden fazla gönderisi olabilir.

Kullanıcı takipçileri vardır

Kullanıcı birden fazla hikaye paylaşabilir

Kulllanıcının engel listesi vardır

Bir kullanıcı bir fazla kişiyle mesajlaşabilir.

Gönderi

Bir gönderinin birden fazla beğeni, yorumu ve etiketi'i olabilir.

Bir kullanıcının gönderilini takip içinde olduğu kullanıcılar tarafından beğenilebilir, yorum yapılabilir ve etiketlenebilir.

Beğeni – Yorum – Etiket

Beğeni, yorum etiket tablolarında kullanıcıların, kullanıcı isimleri mevuttur

Gönderi Tür

Bir gönderinin birden fazla türü olabilir.

Bir gönderi aynı anda hem resim hem video türünde olabilir (çoklu paylaşım ile)

Mesaj Tür

Bir mesajın birden fazla türü olabilir. Mesajlaşma özelliği yine takipçiler arasında gerçekleşebilir

Mesajın grup mesaj ya da direkt mesaj gibi özellikleri vardır

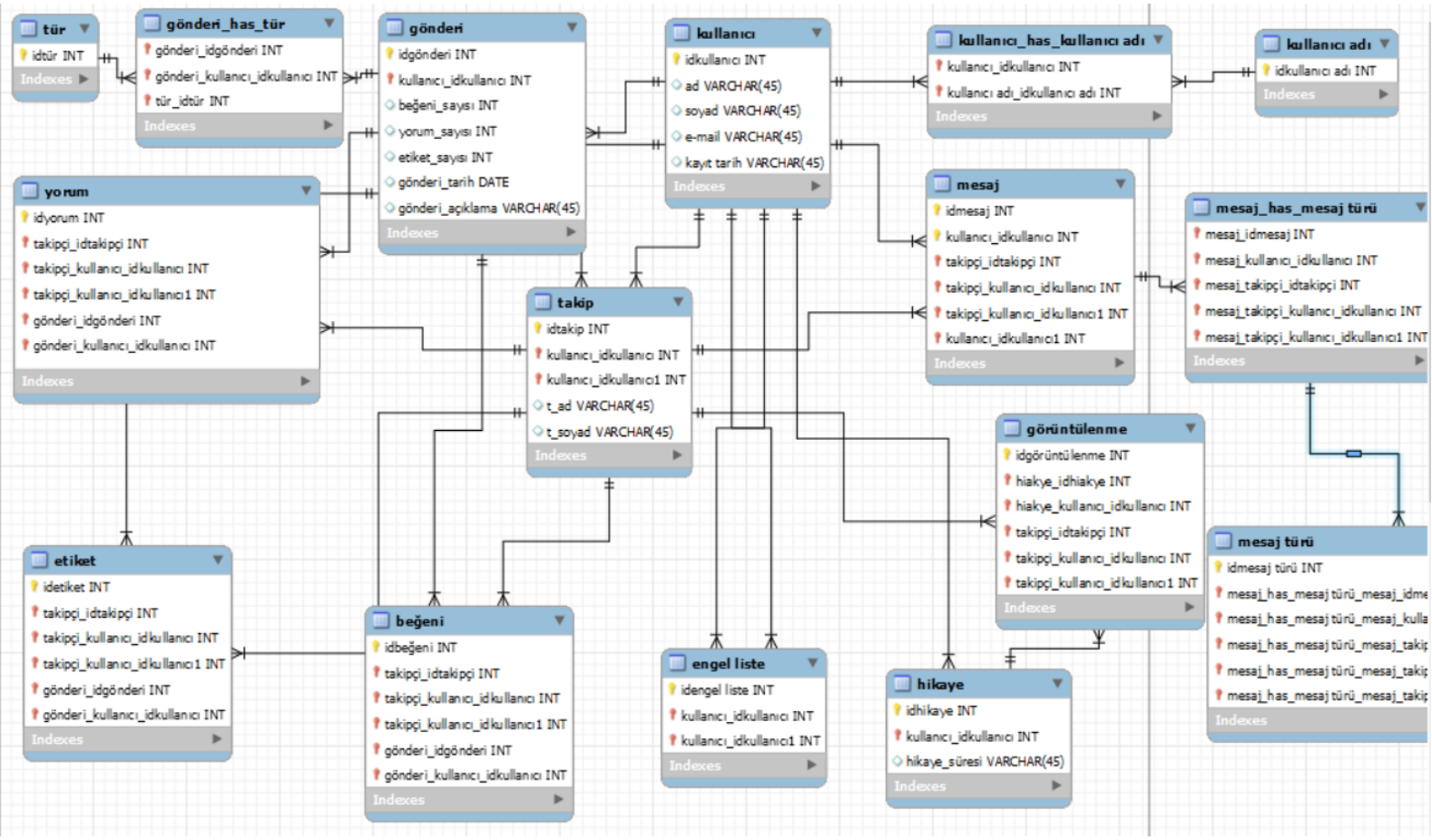
Hikaye - Görüntüleme

Kullanıcı birden fazla hikaye paylaşabilir. Hikaye takipçiler tarafından görüntülenebilir. Bir hikayeyi birden fazla takipçi görüntüleyebilir.

Engel Liste

Kullanıcının engellediği kişilerin kullanıcı adları mevcuttur

2- TABLOLAR



3- 3NF'A KADAR NORMALLEŞTİRME ANALİZİ

Gönderi_id	Gönderi_tür	Gönderi_tarih	Gönderi_açıklama	kullanıcı_İd	kullanıcı_ad	ad,soyad	Kullanıcı_kayıt tarihi	kullanıcı _maill
G1	resim	2020	xxx	K1	mb	MerveBalcı	2015	m@
G2	video	2021	x	K2	ak	AhmetAk	2018	ak@
G3	Resim,video	2022	x	K3	fs	FatmaSarı	2018	fs@
G4	video	2021	xxxxx	K1	mb	MerveBalcı	2015	m@

Bir tablonun birinci normal formda olması için bir kayıttaki tüm alanlar bir tek manaya sahip veri içermelidir. Buradaki tablomuzda gönderi türünde birkaç tür bulunabiliyor

1.NF

Gönderi_id	Gönderi_tür	Gönderi_tarih	Gönderi_açıklama	kullanıcı_İd	kullanıcı_ad	ad,soyad	Kullanıcı_kayıt tarihi	kullanıcı _maill
G1	resim	2020	xxx	K1	mb	MerveBalcı	2015	m@
G2	video	2021	x	K2	ak	AhmetAk	2018	ak@
G3	Resim	2022	x	K3	fs	FatmaSarı	2018	fs@
G3	Video	2022	x	K3	fs	FatmaSarı	2018	fs@
G4	video	2021	xxxxx	K1	mb	MerveBalcı	2015	m@

Gönderi_id → Gönderi_tarih, Gönderi_açıklama, kullanıcı_İd, kullanıcı_ad, ad- soyad, Kullanıcı_kayıt_tarihi ,kullanıc_maill

Kullanıcı_id → kullanıcı_ad, ad- soyad, Kullanıcı_kayıt_tarihi ,kullanıc_maill

Yukarıda kısmi bağımlılıklarımızı bulduk, aday anahtarımız buradan yola çıkarak gönderi_id ve gönderi_tür olur

R= { Gönderi_id, Gönderi_tür, Gönderi_tarih, Gönderi_açıklama, kullanıcı_İd, kullanıcı_ad, ad- soyad, Kullanıcı_kayıt_tarihi,
kullanıc_mail }

Aday anahtarlar → Gönderi_id, Gönderi_tür

2. Normal form için aday anahtarlarına bakarak kısmi bağımlılık var mı, varsa buna göre bölümleme yapmamız gerekiyor.

R1 = { Gönderi_id, gönderi_tarih, kullanıcı_id, gönderiA, ad soyad, kullanıcı_ad, kullanıcı_kayıt tarihi ,mail }

R2 = { Gönderi_id, Gönderi_tür }

Buna göre;

GÖNDERİ TÜR

Gönderi_id	Gönderi_tür
G1	resim
G2	video
G3	resim
G3	video
G4	reels

Gönderi_id	Gönderi_tarih	Gönderi_açıklama	kullanıcı_İd	kullanıcı_ad	Ad	Soyad	Kullanıcı_kayıt tarihi	kullanıcı _maill
G1	2020	xxx	K1	mb	Merve	Balcı	2015	m@
G2	2021	x	K2	ak	AhmetAk	Ak	2018	ak@
G3	2022	x	K3	fs	FatmaSarı	Sarı	2018	fs@
G3	2022	x	K3	fs	FatmaSarı	Sarı	2018	fs@
G4	2021	xxxxx	K1	mb	MerveBalcı		2015	m@

3. Normal form uygulamamız için bu sefer aday anahtarı dışındaki kısmi bağımlılıklarımıza bakmamız lazım

3.NF

R1 = {gönderi id gönderi_tür}

R2 = { Gönderi_id, gönderi_tarih, kullanıcı_id, gönderiA, ad soyad, kullanıcı_ad, kullanıcı_kayıt tarihi ,mail }

R21 = { Gönderi_id, gönderi_tarih, kullanıcı_id, gönderiA }

R22 = { Kullanıcı_id, kullanıcıad soyad, kullanıcı_kayıt tarihi,mail }

Buna göre

GÖNDERİ TÜR

Gönderi_id	Gönderi_tür
G1	resim
G2	video
G3	resim
G3	video
G4	reels

GÖNDERİ

Gönderi_id	Gönderi_tarih	Gönderi_açıklama	kullanıcı_İd
G1	2020	xxx	K1
G2	2021	x	K2
G3	2022	x	K3
G3	2022	x	K3
G4	2021	xxxxx	K1

KULLANICI

kullanıcı_İd	kullanıcı_ad	Ad	Soyad	Kullanıcı_kayıt tarihi	kullanıcı _maill
K1	mb	Merve	Balcı	2015	m@
K2	ak	Ahmet	Ak	2018	ak@
K3	fs	Fatma	Sarı	2018	fs@
K3	fs	Fatma	Sarı	2018	fs@
K1	mb	Merve	Balcı	2015	m@

4- SORGULAR

1- Kullanıcı adı aa olan ve 2021 yılında paylaşılan gönderi türü resim olan, gönderilerinin id'si bulunuz

Sql

SELECT g.id FROM gönderi g INNER JOIN kullanıcı k USING (kullanıcı_ad)

WHERE k.kullanıcı_ad= 'aa' and g.tür = 'resim'

İlişkisel cebir

π gonderi.id (σ gönderi.kullanıcı_ad = kullanıcı.kullanıcı_ad and kullanıcı.kullanıcı_ad 'aa' and kullanıcı.yıl = 2021(gönderi x kullanıcı))

2- Genel beğeni ortalaması üstünde olan kişilerin kullanıcı adlarını bulunuz

Sql

```
select * from
(select AVG(begenı_s) as ort1 from gonderı
 group by kullanıcı_ıd)b,
 (select AVG(begenı_s) as ort2 from gonderı) a
  where b.ort1 > a.ort2
```

İlişkisel cebir

genel = γ ;avg(begeni_s)→genel_ortalama gönderi

kullanıcı = γ kullanıcı_id;avg(begeni_s)→kullanıcı_ortalama gönderi

kullanıcı ⋈ (genel ⋈ kullanıcı_ortalama >genel_ortalama gönderi)

3- **2021 yılında en az bir gönderi paylamış ve beğeni sayısı 30 üstünde olan, fakat 2022 yılında hiç gönderi paylaşmamış kullanıcıların isimlerini**

Sql

select kullanıcı_ıd from kullanıcı k

ınner joın gonderı g usıng(kullanıcı_ıd)

where g.trhh=2021 and g.begenı_s >30

group by kullanıcı_ıd

mınus

select kullanıcı_ıd from kullanıcı k

ınner joın gonderı g usıng(kullanıcı_ıd)

where g.trhh=2022

group by kullanıcı_ıd

İlişkisel cebir

π kullanıcı_ıd (σ kullanıcı.kullanıcı_ıd = gönderi.kullanıcı_ıd and gönderi.tarih = 2021 and gönderi.beğeni_sayı > 30 (kullanıcı x gönderi))

-

π kullanıcı_ıd (σ kullanıcı.kullanıcı_ıd = gönderi.kullanıcı_ıd and gönderi.tarih = 2022 (kullanıcı x gönderi))

Beğeni sayısını beğeni tablosundan, beğenen kullanıcılardan bularak ;

```
select k.kullanıcı_ıd, g.gonderı_ıd, count(b.t_kad) from kullanıcı k
 joın gonderı g on (g.kullanıcı_ıd = k.kullanıcı_ıd)
joın begenı b on (g.gonderı_ıd = b.gonderı_ıd)
where  g.trhh=2021
group by  k.kullanıcı_ıd,g.gonderı_ıd
havıng count(b.t_kad) > =30
mınus
select k.kullanıcı_ıd, g.gonderı_ıd, count(b.t_kad) from kullanıcı k
joın gonderı g on (g.kullanıcı_ıd = k.kullanıcı_ıd)
joın begenı b on (g.gonderı_ıd = b.gonderı_ıd)
where g.trhh=2022
group by k.kullanıcı_ıd, g.gonderı_ıd
```

İlişkisel cebir

π kullanıcı_ıd (kullanıcı ⋈ (σ gönderi.tarih = 2021 and begeni_s > 30 (γ gonderı_ıd ;count(begeni.tk_ad)→ begeni_s gönderi ⋈ begeni)))

-

π kullanıcı_ıd (kullanıcı ⋈ (σ gönderi.tarih = 2022 (γ gonderı_ıd ;count(begeni. .tk_ad)→ begeni_s gönderi ⋈ begeni)))

5- PL SQL fonksiyonu

Sosyal medya hesabı olan her kullanıcı etkileşim oranına sahiptir, etkileşim oranları kullanıcıların, izlenme, beğenilme, yorum gibi durumlara göre farklılık göstermektedir. Kullanıcıların paylaştıkları gönderilerden, oluşturduğu etkileşimi hesaplamak için fonskiyon yazdım

```
--  DDL for Package HESAP_PACK
--------------------------------------------------------

  CREATE OR REPLACE PACKAGE "HR"."HESAP_PACK" AS

  /* TODO enter package declarations (types, exceptions, methods etc) here */
  FUNCTION GÖRÜNTÜLENME_HESAP(yorum PLS_INTEGER,beğeni PLS_INTEGER, gönderi_turu VARCHAR2) RETURN NUMBER;
  FUNCTION GÖRÜNTÜLENME_HESAP(yorum PLS_INTEGER,beğeni PLS_INTEGER, izlenme_sayısı, gönderi_turu VARCHAR2) RETURN NUMBER;

END HESAP_PACK;

--------------------------------------------------------
--  DDL for Package Body HESAP_PACK
--------------------------------------------------------

  CREATE OR REPLACE PACKAGE BODY "HR"."HESAP_PACK" AS
    FUNCTION GET_RESİM_GÖRÜNTÜLENME(beğeni_adet PLS_INTEGER,yorum_adet PLS_INTEGER) RETURN NUMBER;
    FUNCTION GET_VİDEO_GÖRÜNTÜLENME (beğeni_adet PLS_INTEGER,yorum_adet, izlenme_sayısı) RETURN NUMBER;
    FUNCTION GÖRÜNTÜLENME_HESAP(yorum PLS_INTEGER,beğeni PLS_INTEGER, gönderi_turu VARCHAR2)  RETURN NUMBER AS
        p_sonuc NUMBER(10,2):=0;
    BEGIN
        IF gönderi_turu ='RESİM' THEN
            p_sonuc:= GET_RESİM_GÖRÜNTÜLENME (beğeni_adet,yorum_adet);
        END IF;

        RETURN p_sonuc;
    END;

    FUNCTION GÖRÜNTÜLENME_HESAP(yorum PLS_INTEGER,beğeni PLS_INTEGER, izlenme_sayısı, gönderi_turu VARCHAR2)  RETURN NUMBER AS
        p_sonuc NUMBER(10,2):=0;
    BEGIN
        IF gönderi_turu ='RESİM' THEN
            p_sonuc:= GET_RESİM_GÖRÜNTÜLENME (beğeni_adet,yorum_adet);
        END IF;

        RETURN p_sonuc;
    END;

    FUNCTION GÖRÜNTÜLENME_HESAP(yorum PLS_INTEGER,beğeni PLS_INTEGER, izlenme_sayısı, gönderi_turu VARCHAR2)  RETURN NUMBER AS
        p_sonuc NUMBER(10,2):=0;
    BEGIN

        IF gönderi_turu ='VİDEO' THEN
          p_sonuc:= GET_VİDEO_GÖRÜNTÜLENME (yorum,beğeni, izlenme_sayısı);
        END IF;
        RETURN p_sonuc;
    END;
    FUNCTION GET_RESİM_GÖRÜNTÜLENME (beğeni_adet PLS_INTEGER,yorum_adet PLS_INTEGER) RETURN NUMBER AS
    BEGIN
        RETURN  beğeni_adet* yorum_adet;
    END;

    FUNCTION GET_VİDEO_GÖRÜNTÜLENME (beğeni_adet PLS_INTEGER,yorum_adet, izlenme_sayısı) RETURN NUMBER AS
    BEGIN
        RETURN  (beğeni_adet* yorum_adet* izlenme_sayısı);
    END;

END HESAP_PACK;
```

6- PL SQL tablo trigger

Kullanıcının paylaştığı gönderilerde etiketleme özelliği var, bu trigger'da bir gönderide 100'den fazla kişinin etiketlenmesini önlemek amacıyla yazılmıştır. Önce etiket sayısını hesaplayan fonksiyon ardından da buradaki fonksiyonu kullanarak trigger oluşturulmuştur.

```
create or replace PACKAGE GNDR_PACK AS
    FUNCTION get_etiket_adet(p_gönderi_id IN PLS_INTEGER) RETURN PLS_INTEGER;
END GNDR_PACK;

create or replace PACKAGE BODY GNDR_PACK AS

  FUNCTION get_etiket_adet(p_gönderi_id IN PLS_INTEGER) RETURN PLS_INTEGER AS
    CURSOR c_etiket_adet IS
        SELECT COUNT(*) FROM etiket
        WHERE gönderi_id=p_gönderi_id;
    p_adet PLS_INTEGER:=0;
  BEGIN
    OPEN c_etiket_adet;
    FETCH c_etiket_adet INTO p_adet;
    CLOSE c_etiket_adet;
    RETURN p_adet;
  END get_etiket_adet;

END GNDR_PACK;

CREATE OR REPLACE TRIGGER trg_etiket_adet_kontrol
BEFORE INSERT ON etiket
FOR EACH ROW
DECLARE
    p_adet PLS_INTEGER;
BEGIN
    p_adet:=gndr_pack.get_etiket_adet(:new.gönderi_id);
    IF p_adet+1>10 THEN
        RAISE_APPLICATION_ERROR(-20555,'Hiç bir gönderide 10''dan fazla kişi etiketlenemez. Mevcut: '||p_adet);
    END IF;
END;
```

7- DENORMALİZASYON

Denormalizasyon, veritabanı yöneticilerinin bir veritabanı altyapısının performansını artırmak için kullandığı bir stratejidir. Çeşitli tablolardaki verileri tek bir tabloda birleştiren veritabanı sorgularıyla ilgili belirli türdeki sorunları azaltmak için normalleştirilmiş bir veritabanına yedekli veri eklemeyi içerir.
Yapılan normalizasyon işlemleri ile ciddi hız kayıpları ortaya çıkabilir. Bu durum performansın düşmesine sebep olabilir. Bu durumda bazı durumları basitleştirmek için tekrar eden verinin tekrar sisteme eklenmesidir. Normalizasyon kuralının tersi olarak düşünülebilir.

Tablolar arasında sorgular yazarken fazla join işlemlerini önlemek ve karmaşıklığı azaltmak için denormalizasyon yapılabilir. Dezavantajı, eğer tablolar büyükse, masalarda birleştirme yapmak için gereksiz yere uzun zaman harcayabiliriz.

Kendi veritabanı ve tablolarıma bakarsak gönderi faaliyetleri bir tabloda toplanabilir yani, gönderi, beğeni tablosu, yorum, gönderi türü ve etiket tabloları, gönderiye ait bilgiler içerdiği için karmaşık sorgu yapılarından kaçınmak için hepsi bir tabloda toplanabilir. Bu şekilde fazla, karmaşık sorgu işlemlerinden kaçınılabilir.